Application Report
ZHCA594 – January 2014

基于 KeyStone 器件建立鲁棒性系统

冯华亮，向展, 卢璟，尹志新

摘要

对于复杂的系统，鲁棒性是非常重要的。为了协助客户建立鲁棒性系统，KeyStone 器件提供了多种硬件保护机制，如内存保护、EDC。本文介绍如何利用这些特性在 KeyStone 器件上建立一个鲁棒的系统。同时提供了与文档配套的例程。

内容

框图

表格

1 简介

如图 1 所示，KeyStone 器件提供了多种协助客户建立鲁棒性应用的特性。

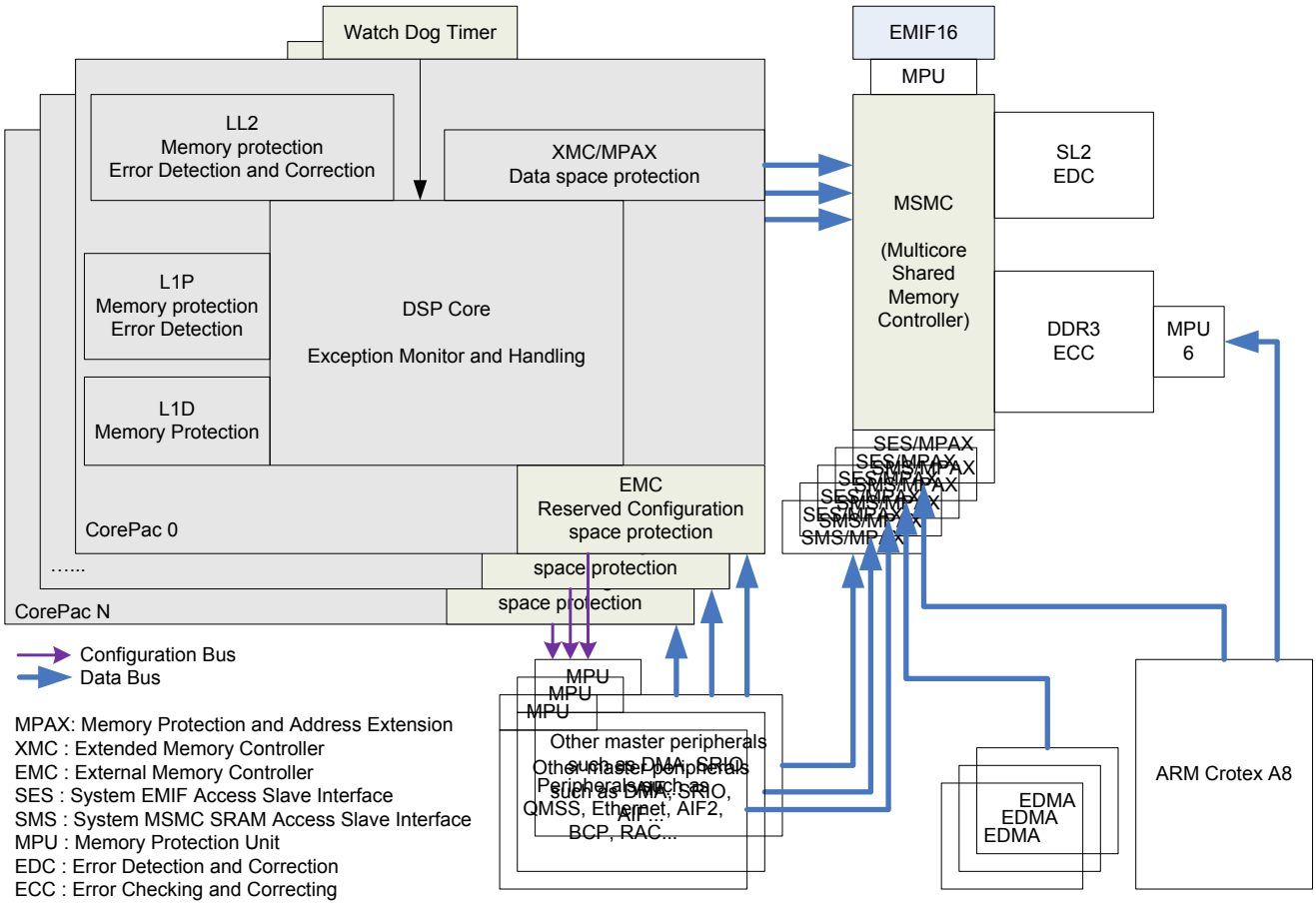

图 1 KeyStone 器件鲁棒系统

如图 1 所示，在 LL2、L1P 及 L1D 中集成了内存保护模块；LL2、SL2 及 DDR 控制器中集成了错误检查纠正模块；L1P 集成了错误检测模块。

MPAX 和 MPU 模块附在总线上，用于监控检测以避免非法的总线访问。

每个 DSP CorePac 有一个独立的 MPAX 用于监控与 MSMC 连接的总线。

对于系统中其他的 master，根据权限 ID 进行分类。对每个权限 ID，在 MSMC 中集成了 2 个 MPAX 用于监视与该权限 ID 相关的访问。其中一个是 SES MPAX 用于保护对 DDR3 的访问，另一个是 SMS MPAX 用于保护对 SL2 的访问。关于每个 master 对应的权限 ID，参考相应的器件手册。

某些外设的配置端口上添加了 MPU，用于保护对该外设配置区域的非法访问。但是并非所有的外设都受 MPU 的保护，具体参考相应器件手册中受 MPU 保护的外设列表。

每个 CorePac 有一个看门狗定时器用于监视其活动，如果该核死机，看门狗可以触发不可屏蔽中断或者复位信号。

EMC 可以避免 DSP core 访问没有映射的的配置空间，XMC 则可以避免 DSP core 访问没有映射的数据空间。

所有这些功能都由硬件模块实现，使用这些功能对系统性能基本上没有影响。使用 EDC 会对存储器的访问性能稍有影响，但从整个系统层面上看，它几乎是微不足道的。

在出现问题时，所有这些模块可以向 DSP core 触发异常，DSP core 的异常监控模块可以记录这些状态并触发异常服务程序执行相应的操作。

本文讨论这些特性的应用，并给出相关基于寄存器层 CSL 实现的例程。代码使用如下方式定义寄存器指针。

```
#include <cslr_cgem.h>
#include <cslr_xmc.h>
#include <cslr_msmc.h>
CSL_CgemRegs * gpCGEM_regs = (CSL_CgemRegs *)CSL_CGEM0_5_REG_BASE_ADDRESS_REGS;
CSL_XmcRegs * gpXMC_regs = (CSL_XmcRegs *) CSL_XMC_CONFIG_REGS;
CSL_MsmcRegs * gpMSMC_regs = (CSL_MsmcRegs *)CSL_MSMC_CONFIG_REGS;
```

上述各种特性具体描述分布于各自子系统的文档中，本文最后的参考章节中列出了所有相关的文档。在看本文之前，假设客户已经阅读了相关属性对应的文档，所以本文旨在提供相关的补充信息。

本文适用于 KeyStone 1 系列 DSP，例程在 TCI6614 EVM，C6670 EVM，C6678 EVM 上进行了验证。对于其他的 KeyStone 器件包扩 KeyStone 2 系列，基本功能都是一样的，一些细节上的些许差异请参阅相应器件手册。

2 内存保护

文档“Memory Protection On KeyStone Devices (SPRWIKI9012)”中讨论了 KeyStone 器件上的内存保护属性，其中包括其它文档中没有的很多有用信息，本节在其基础上做一些总结和补充。

表 1 总结列出不同内存保护模块的差异。

表 1 各种内存保护模块

内存保护模块	系统中的位置	page/segment/range		备注
		Number	Size	
L1D 内存保护	Slave port	16	2KB	每个 CorePac 有独立的 L1D,L1P,LL2 及 XMC.
L1P 内存保护	Slave port	16	2KB	
LL2 内存保护	Slave port	32	(LL2 size)/32	
XMC MPAX	Master port	16	Programmable	
SMS MPAX	Master port	8	Programmable	每个权限 ID 有一组 SMS 及 SES.
SES MPAX	Master port	8	Programmable	
MPU	Slave port	1~16	Programmable	不同器件的 MPU 对应不同的可配范围及内存宽度。

系统中有多个 master 和 slave，位于 slave 输入端口的保护模块用于阻止来自其他 master 对该 slave 的非法访问；位于 master 输出端口的保护模块用于阻止该 master 对所有其他 slave 的非法访问。

每个内存页、分片或范围的保护属性都是可编程的。

2.1 L1 及 LL2 内存保护

关于 L1 及 LL2 内存保护的基本信息参考“TMS320C66x CorePac User Guide(SPUGW0)”中内存保护章节。

L1 及 LL2 内存保护只区分 7 个外部请求 ID，但是系统可能有 16 个权限 ID。默认情况下，系统权限 ID 0~5 映射到 CorePac AID 0~5，所有其他的权限 ID 均映射到 AIDx。

CorePac AID 与系统权限 ID 之间的映射关系可由 EMC 编程配置，具体参考“TMS320C66x CorePac User Guide(SPUGW0)”中“外部存储控制器(EMC)”章节。

注意，IDMA 的 AID 与其所属 CorePac 的数值一致，EDMA 传输的权限 ID 与配置并发起这个传输的核的编号一致。

通常 L1 被配置为 cache，此时所有 L1 相关的内存保护属性寄存器应该清零从而阻止其他 master 的对 L1 的访问。

CorePac 内部内存保护模块（保护 L1，LL2 及 XMC/MPAX）的寄存器被一个锁保护起来。默认情况下，这些寄存器没有被锁住，用户软件可以使用自定义的密钥锁住这些寄存器，然后，只有用该密钥进行解锁后才可以访问这些寄存器。

2.2 共享内存保护 – MPAX

关于 CorePac 共享内存保护的基本信息参考“TMS320C66x CorePac User Guide(SPRUGW0)”中“扩展存储控制器(XMC)”章节；关于系统中其他 master 的共享内存保护基本信息参考“KeyStone Architecture Multicore Shared Memory Controller User Guide (SPRUGW7)”中“内存保护及地址扩展 (MPAX)”章节。

如下是例程中关于 XMC/MPAX 的配置样例，每一行代表 MPAX 中的一个分片配置。

```
MPAX_Config XMC_MPAX_cfg_table[]=
{
    /*BADDR         RADDR           SegementSize      AccessPermisionMask
    32-bit virtual  36-bit physical in byte, must     Access types allowed
    base address    address right   be power of 2     in this address range
                    shift by 4                                          */
    {0x0C000000,    0x00C000000>>4,  2*1024*1024,    MP_SR|MP_SW|MP_SX|MP_UR|MP_UW|MP_UX},/*SL2, RWX*/
    {0x10000000,    0x010000000>>4,  0x04000000,     MP_SR|MP_SW|MP_UR|MP_UW},/*LL2 global, RW*/
    {0x18000000,    0x00C100000>>4, 1*1024*1024,    MP_SR|MP_SW|MP_UR|MP_UW},/*SL2 remap */
    {0x20000000,    0x020000000>>4,  0x04000000,     MP_SR|MP_SW|MP_UR|MP_UW},/*peripherals*/
    {0x30000000,    0x030000000>>4,  0x08000000,     MP_SR|MP_SW|MP_UR|MP_UW},/*peripherals*/
    {0x40000000,    0x040000000>>4,  0x10000000,     MP_SR|MP_SW|MP_UR|MP_UW},/*HyperLink*/
    {0x60000000,    0x060000000>>4,  0x10000000,     MP_SR|MP_SW|MP_UR|MP_UW},/*PCIE*/
    {0x21000000,    0x100000000>>4,  4*1024,         MP_SR|MP_SW|MP_UR|MP_UW},/*DDR3 config*/
    {0x80000000,    0x800000000>>4,  0x10000000,     MP_SR|MP_SW|MP_SX|MP_UR|MP_UW|MP_UX},/*DDR3*/
    {0x90000000,    0x810000000>>4,  0x10000000,     MP_SR|MP_SW|MP_UR|MP_UW}/*DDR3*/
};
```

逻辑地址低于 0x0C00_0000 的地址访问不会进入 XMC。对地址空间 0x0000_0000~0x07FF_FFFF 进行访问时，在 C66x CorePac 内部进行地址解析。这块地址范围包括内部及外部配置总线，及 L1D、L1P、L2 存储空间。

对位于 0x0C00_0000~0x0FFF_FFFF 区间的逻辑地址访问时，会经过 L1 cache，并且在读操作时会经过预取缓存，与该地址范围对应的内存属性配置寄存器 MAR 是硬件拉死的，不可修改。也就是说对该逻辑地址空间的访问在进入 XMC MPAX 之前不会经过 L2 cache，所以这块逻辑地址空间称为“快速 SL2 RAM 路径”。

对大于等于 0x1000_0000 的逻辑地址访问会首先经过 L2 cache 控制器，然后经过 XMC MPAX，这种常规路径会增加一个 cycle 的时延。

根据上述配置例子，在访问 SL2 时，采用逻辑地址 0x0C00_0000 的访问速率高于使用重映射后的逻辑地址 0x1800_0000。但是 0x1800_0000 对应的内存属性寄存器 MAR 是可编程的，因此可以配置通过 0x1800_0000 访问的 SL2 为 non-cacheable 及 non-prefetchable。

注意替换地址 RADDR 是 36-bit 物理地址>>4。常见的 DDR3 错误配置如下：

```
    {0x80000000,    0x800000000>>4,  0x10000000,     MP_SR|MP_SW|MP_SX|MP_UR|MP_UW|MP_UX},/*DDR3*/
    {0x90000000,    0x900000000>>4,  0x10000000,     MP_SR|MP_SW|MP_UR|MP_UW}/*DDR3*/
```

或者

```
    {0x80000000,    0x80000000,  0x10000000,     MP_SR|MP_SW|MP_SX|MP_UR|MP_UW|MP_UX},/*DDR3*/
    {0x90000000,    0x90000000,  0x10000000,     MP_SR|MP_SW|MP_UR|MP_UW}/*DDR3*/
```

注意 DDR3 起始物理地址为 0x8:0000_0000，而 0x9:0000_0000 相对起始地址有 4GB 的偏移，在大多数系统中这是一个非法的地址。

在真实系统中，应该充分利用好 MPAX 的所有片段更好地将存储空间划分成尽可能多的小片，并仔细设定各个分片的访问限定属性。

不用的地址不应该映射，MPAX 会拒绝对未映射的地址访问并上报异常事件，从而有助于捕获软件错误。

SMS/MPAX 只允许对 SL2 的访问，如下为其配置例子：

```
MPAX Config SMS MPAX cfg table[]=
{
    /*BADDR         RADDR           SegementSize      AccessPermisionMask
    32-bit virtual  36-bit physical in byte, must     Access types allowed
    base address    address right   be power of 2     in this address range
                    shift by 4                                          */
    {(Uint32)msmcSmsMPDataDst, 0,  sizeof(msmcSmsMPDataDst), MP SW|MP UW},/*SL2 write only data*/
    {(Uint32)msmcSmsMPDataSrc, 0,  sizeof(msmcSmsMPDataSrc), MP_SR|MP_UR},/*SL2 read only data*/

};
```

SES/MPAX 是用于保护对 DDR3 的访问，如下为其配置例子：

```
MPAX_Config SES_MPAX_cfg_table[]=
{
    /*BADDR         RADDR           SegementSize      AccessPermisionMask
    32-bit virtual  36-bit physical in byte, must     Access types allowed
    base address    address right   be power of 2     in this address range
                    shift by 4                                          */
    {(Uint32)msmcSesMPDataDst, 0,  sizeof(msmcSesMPDataDst), MP_SW|MP_UW},/*SL2 write only data*/
    {(Uint32)msmcSesMPDataSrc, 0,  sizeof(msmcSesMPDataSrc), MP_SR|MP_UR},/*SL2 read only data*/
};
```

当两个 master 通过共享 memory 交换数据时，应该确保两个 master 使用的逻辑地址映射到相同的物理地址。

注意 EDMA 的权限 ID 是继承于对其配置的 CorePac。

警告：

在修改一条 MPAX 表项时，需要确保此时没有对该表项所覆盖地址的访问。在修改之前，需要先将该表项覆盖地址对应的 cache 及预取缓存中的数据进行回写及失效操作。

对于 MPAX 的配置，推荐在程序开始之初且没有使用任何共享存储空间之前完成。用于 CorePac MPAX 配置的代码和数据应该放在 LL2。

如果要运行时动态修改一个 MPAX 表项，安全的方法是先将新的配置写到一个未使用的编号高度表项，然后清掉旧的表项。这是由于编号高度表项的优先级高于编号低端表项。

在修改 MPAX 表项之前需要先执行如下操作：

1．将 MPAX 表项对应的存储空间内容从 cache 中剔除出去。即使对于属性为不可写的存储空间，应该使用 CACHE_wbInvL2()而非 CACHE_inv L2（）。

2．如果对受影响的存储器空间使能了预取功能，则需要对预取缓存执行失效操作。

3．执行“MFENCE”确保回写及失效操作完成。

CorePac 的 MPAX 寄存器受 CorePac 的内存保护寄存器锁保护。SES 及 SMS 的 MPAX 内存保护属性寄存器被 MSMC 内部分别用于 SES 及 SMS 的锁保护。MSMC 内部其他寄存器被 MSMC 内部用于非 MPAX 的锁保护。

2.3 外设配置端口保护 – MPU

关于 MPU 的基本信息参考“KeyStone Architecture Memory Protection Unit User Guide (SPRUGW5)”。

MPU0、MPU1、MPU2 及 MPU3 对所有 KeyStone 1 器件是相同的。但是对于不同的器件，其附加 MPU 的个数，每个 MPU 支持的地址范围表项数，MPU 的默认配置均有所差异。具体可参考相关器件手册的“内存保护单元(MPU)”章节。

MPU 与 MPAX 的区别在于，如果访问地址不在 MPU 任何一个地址范围内，则该地址访问是允许的；而当该地址与 MPAX 中任意表项地址范围不匹配时，则该地址访问被拒绝。

注意，如果没有被 MPPA 的设置所拒绝，MPU 单元默认所有的访问都是许可的。对于一个地址访问，MPU 首先将访问的权限 ID 与 MPPA 寄存器的 AID bit 配置进行核对，如果与权限 ID 对应的 AID bit 为 0，则不需要核对地址范围，该访问被许可。如 MPPA=0 则允许所有的对该空间的访问，如果要拒绝任意对该空间的访问则需要将 MPPA 配置为 0x03FFFC00。L1 及 LL2 内存保护的 MPPA 设置则有所不同，当 MPPA 中 AID bit 为 0 是拒绝相应的访问。

当传输与 MPU 中多个地址范围匹配时，所有重叠的范围必须允许其访问，否则该访问会被拒绝。最终赋予的访问权限与所有匹配表项中最低的权限等级一致。如某传输与 2 个表项匹配，其中一个是 RW，另一个是 RX，则最终的权限是 R。这与 MPAX 也是不一样的。如果一个地址落入多个 MPAX 表项，编号高的表项优先于编号低的表项。MPAX 只会用编号最高的表项决定权限，并忽略其他匹配的表项。

如下与本文对应例程中一个对 MPU1 的配置例子。每行代表 MPU 中一个配置范围。

```
MPU Range Config MPU1 cfg table[]=
{
  /*StartAddr  EndAddr     AccessPermisionMask*/
  {0x34020000, 0x3403FFFF, MP_AID11|MP_NS|MP_EMU|MP_SR|MP_UR}, /*Queue 0~8191, deny write from AID11*/
  {0x34020000, 0x34027FFF, MP_AID8_15|MP_NS|MP_EMU|MP_SR|MP_UR},/*Queue 0~2047,deny write from AID8~15*/
  {0x34038000, 0x3403FFFF, MP_AID0_7|MP_NS|MP_EMU|MP_SR|MP_UR},/*Que 6144~8191,deny write from AID0~7*/
};
```

如上配置知，队列保护如下：

- 队列 0~2047 只可由 AID0~7 进行写（PUSH）操作；
- 队列 2048~6143 可由 AID11 以外所有的 AID 进行写（PUSH）操作；
- 队列 6144~8191 只可由 AID8~15(AID11 除外)进行写（PUSH）操作。

TCI6614 上的 MPU6 用于避免 ARM 对 DDR3 的非法操作。注意，MPU6 是用于低 32-bit DDR 物理地址范围的保护。

注意，为了清除 MPU 异常/中断事件，必须在服务程序的最后向 EOI 寄存器写 0.

TCI6614 的 MPU 事件与其他 KeyStone 器件有所不同。TCI6614 中所有的 MPU0~7 事件被合并为一个事件并作为一个系统事件连接到 CIC0。由于 TCI6614 MPU 事件是电平中断而非脉冲中断事件，所有必须首先清除 MPU 事件标志，然后才可以清 CIC 标志。对于脉冲中断事件，必须首选清 CIC 标志，然后清源标志。

另外，只有在通过 PSC 使能 BCP 后，才可以访问 TCI6614 中用于 BCP 的 MPU5。即在访问 TCI6614 中 MPU5 寄存器时，如果此时 BCP 没有被使能，则该访问将触发访问错误。

2.4 预留区域保护

预留区域（非法地址）被自动保护。对非法地址进行读操作时将返回垃圾数据，写操作则会被阻止。对预留区域的访问可以产生异常，这有益于捕获软件 bug。

由于 DSP core 的访问会经过 L1D 控制器，所以 DSP core 对非法地址的访问会触发 L1D 内存保护异常。

DSP core 从非法地址执行时将触发指令获取异常。

对于非法写操作，触发的异常取决于相应的目的地址。

DMA 对非法地址访问时，DMA 模块会上报总线错误。DMA 错误事件可以作为异常路由到 DSP core。

3 EDC

EDC（Error Detection and Correction）用于存储器软错误(Soft Error)。软错误是一个错误的信号或数据，但是并不意味着硬件被破坏。在观测到一个软错误后，并不意味着系统可靠性会下降。在宇宙飞船中这种类型的错误称为单一事件扰乱。在内存系统中，一个软错误会改变程序中的一条指令或者一个数据值。软错误通常可以通过器件的重启进行纠正，而硬件错误通常不能通过重启来恢复。软错误不会对系统硬件造成破坏；仅仅会对处理的代码或数据造成错误。产生软错误的原因有：

1. 阿尔法粒子辐射及宇宙射线产生能量中子及质子。发生的概率取决于器件的地理位置及周围环境。通常，一个器件在几年中才会出现几次。
2. 软错误也可由随机噪声、干扰或信号完整性错误引发，如板载电感应或电容串扰。如果软错误发送概率高于上述条目 1 中的理论值，则应该检查硬件设计找出其他原因。一个常见的原因是供电电源电压低于预期，导致器件对噪声或干扰的影响更敏感。

KeyStone 器件各级 memory 中都实现了 EDC 机制，下表对不同 memory 模块的实现机制进行了比较。

表 2 KeyStone memory 中实现的 EDC 机制

memory	error detection	Error correction	Segment line size
L1P	1 bit	N/A	64 bits
CorePac L2	2 bits	1 bit	128 bits
Shared L2 (MSMC)	2 bits	1 bit	256 bits

3.1 L1P 错误检测

关于 L1P 及 LL2 EDC 基本信息参考“TMS320C66x DSP CorePac User Guide(SPRUGW0)”。

校验比特生成与核对：校验比特在进行 64-bit 对齐的 DMA 写或 L1P cache 缓存时生成。非 64-bit 对齐的 DMA 访问将使校验信息失效。在 256-bit 对齐的程序读取或 64-bit 对齐的 DMA 读操作时，L1P EDC 逻辑会核对校验信息。

错误检查设置：器件复位后默认情况下 L1P 错误检查特性是关闭的。一旦 L1PEDCMD 寄存器中的“EN”bit 被置位，所有 L1P memory 中的 ED 逻辑被使能。下面是从应用代码中摘录的 L1P ED 功能使能例子。

```
void L1P_EDC_setup()
{
  Uint32 preL1PMPPA[16];

  /* 1. Disable the EDC */
  CSL_CGEM_disablePMCErrorDetection();

  /* 2. Clear any EDC errors */
  CSL_CGEM_clearPMCErrorDetectionStatus(1, 1);

  /* 3. Memory Scrubbing with IDMA, generate the parity bits*/
```

```
  memcpy(preL1PMPPA, (void *)gpCGEM_regs->L1PMPPA, 64);//save protection attributes
  L1P_memory_protection_cfg(0xFFFF); //enable IDMA access to L1P
  IDMA_copy(0x00E00000, 0x00E00000, 32*1024, DMA_WAIT);
  L1_MPPA_setup(gpCGEM_regs->L1PMPPA, preL1PMPPA);//restore protection for L1

  /* 4. Enable the EDC*/
  CSL_CGEM_enablePMCErrorDetection();
}
```

注意：要使 L1P ED 功能工作正常，必须同时使能 L2 EDC。

对 L1P cache 访问时的错误处理： 对从 L1P cache 中获取程序产生的校验错误，没有专用的系统事件，然而，错误检测逻辑会发送一个直接的异常事件给 DSP（IERR.IFX 事件），然后用户可以使用内部异常事件获取这个错误。L1PEDSTAT 寄存器的 PERR bit 会被置位。L1PEDARRD 寄存器会记录包含错误 bit 的的地址信息。在 L1P 错误对应的异常处理服务函数中，需要对包含错误地址的 cache line 进行失效操作。

对 DMA 访问的错误处理：对 DMA/IDMA 访问产生的校验错误，对应#113 号系统事件。用户可以使用这个事件获取错误。L1PEDSTAT 寄存器的 DERR 比特位会被置位，并且 L1PEDARRD 寄存器会记录包含错误 bit 的地址信息。

L1P EDC 功能验证：通过置位 LPEDCMD 寄存器中的 SUSP 比特可以暂停 L1P EDC 逻辑。使用该特性，可以软件模仿 EDC 错误并验证 EDC 功能。与本文对应的例程中提供了验证 L1P EDC 功能的代码，对应函数 L1P_ED_test()。

3.2 LL2 错误检查与纠正

校验比特生成与核对：在对 L2 以 128 bits 为单元进行内存写操作时会产生相应的校验信息。非 128-bit 对齐或者小于 128 bits 的写操作会使校验信息失效。对 128-bit 对齐的 memory 读操作时，LL2 EDC 逻辑会核对校验信息。更多信息参考“TMS320C66x DSP CorePac User Guide（SPRUGW0）”。

错误检查及纠正配置：器件复位后默认情况下 LL2 EDC 特性是被关闭的。与某些 C64+ DSP 不同的是，KeyStone DSP 不能对内存分块使能 EDC。一旦 EDC 使能，EDC 逻辑对整个 CorePac L2 内存生效。然而，可以对不同的内存访问请求者分别使能，如 L1D 控制器、L1P 控制器或 DMA 控制器。例如，如果用户只需要对代码段使用 EDC，需要使能下面三个域：

1 设置 L2EDCMD 寄存器中的 EN bit 以使能 LL2 EDC 逻辑；

2 设置 L2EDCEN 寄存器中的 PL2SEN 比特以使能 L1 SRAM 的 EDC 逻辑对 L1P 访问的检查；

3 设置 L2EDCEN 寄存器中的 PL2CEN 比特以使能 L2 cache 的 EDC 逻辑对 L1P 访问的检查。

从关闭到使能状态转变时，LL2 EDC 逻辑不会初始化校验 RAM。因此，在进入使能状态后，校验 RAM 中的值是随机值，需要用户软件对其进行初始化，。对 L2 EDC 的配置必须遵循“TMS320C66x DSP CorePac User Guide(SPRUGW0)”中阐述的 EDC 配置顺序。下面是从例程中摘录的 L2 EDC 使能函数参考代码：

```
void LL2_EDC_setup()
{
  int i;
```

```
  unsigned int uiByteCnt= 512*1024;
  TDSP_Board_Type DSP_Board_Type;

  /* 1. Disable the EDC */
  CSL_CGEM_disableL2EDC();

  /* 2. Clear any EDC errors */
  CSL_CGEM_clearL2EDCErrorStatus(1, 1, 1, 1);

  /* 3. Memory Scrubbing with IDMA, generate the parity bits*/
  DSP_Board_Type = KeyStone_Get_dsp_board_type();
  if((DUAL_NYQUIST_EVM == DSP_Board_Type)
    ||(C6670_EVM == DSP_Board_Type)
    ||(TCI6614_EVM == DSP_Board_Type))
  {
    uiByteCnt= 1024*1024;
  }

  /*Each IDMA can transfer up to 65532 bytes,
  here we transfer 32KB each time*/
  for(i=0; i< (uiByteCnt>>15); i++)
  {
    IDMA_copy((0x00800000 + i*(1<<15)), (0x00800000 + i*(1<<15)),
      (1<<15), DMA_WAIT);
  }

  /* 4. Enable the EDC*/
  CSL_CGEM_enableL2EDC();
  gpCGEM_regs->L2EDCEN= (1<<CSL_CGEM_L2EDCEN_DL2CEN_SHIFT)
    |(1<<CSL_CGEM_L2EDCEN_DL2SEN_SHIFT)
    |(1<<CSL_CGEM_L2EDCEN_PL2CEN_SHIFT)
    |(1<<CSL_CGEM_L2EDCEN_PL2SEN_SHIFT)
    |(1<<CSL_CGEM_L2EDCEN_SDMAEN_SHIFT);

}
```

对来自 L1D 控制器的访问错误处理：在经过 L1D cache 从 LL2 中获取数据时，对所有这些数据会进行错误检查，但是不会有任何的纠正。不管是 1-bit 或者是多 bit 错误，将会通过#117 号系统事件（L2_ED2：不可纠正比特错误检测）上报给 DSP core。

对来自 L1P 及 DMA 控制器的访问错误处理：1-bit 错误可以被纠正并通过#116 号系统事件（L2_ED1：可被纠正的比特错误）上报 。2-bit 错误可以被检测，并通过#117 号系统事件上报该错误。

下表列出对于不同存储器访问请求者，相应的 1-bit 错误处理细节。

表 3 LL2 EDC 1-bit 错误处理

Requester	L1D	L1P	DMA
Status register (L2EDSTAT)	“DERR” field = 1 “ NERR” field : no update “BITPOS” field: all zero	“PERR” field = 1 “ NERR” field = 1 “BITPOS” field: error bit position	“DMAERR” field = 1 “ NERR” field = 1 “BITPOS” field: error bit position

Error address (L2EDADDR)	Error address is recorded	Error address is recorded	Error address is recorded
Correctable error counter L2EDCPEC	No update	Increase by 1	Increase by 1
Non-correctable error counter L2EDNPEC	Increase by 1	No update	No update

错误计数器（L2EDCPEC, L2EDNPEC）非常有用，可用于在长时间运行的系统中评估校验比特错误发生的种类与概率。

下表列出对不同存储器访问请求者，相应的 2-bit 错误处理细节。

表 4 LL2 EDC 2-bit 错误上报

Requester	L1D	L1P	DMA
Status register (L2EDSTAT)	“DERR” field = 1 “ NERR” field : no update “BITPOS” field: all zero	“PERR” field = 1 “ NERR” field = 2 “BITPOS” field: all zero	“DMAERR” field = 1 “ NERR” field = 2 “BITPOS” field: all zero
Error address (L2EDADDR)	Error address is recorded	Error address is recorded	Error address is recorded
Correctable error counter L2EDCPEC	No update	No update	No update
Non-correctable error counter L2EDNPEC	Increase by 1	Increase by 1	Increase by 1

对于大于 2 bits 的错误，EDC 逻辑可能会检测并报告为 1-bit 或 2-bit 错误，或者 EDC 根本检测不到该错误。所以说，KeyStone 系列 EDC 硬件逻辑只能保证检测 2-bit 错误或纠正 1-bit 错误。

通常软错误出现的概率很低，首先出现 1-bit 错误，在相对长时间后，第二个错误 bit 也许会产生。由于 1-bit 错误可以被纠正，而 2-bit 错误不能被纠正，所以我们应该尽可能在第二个比特错误出现前纠正好第一个比特错误。

纠正 1-bit 错误的操作通常称为“刷新”。 为了刷新一块存储器，可以使用 IDMA，把 IDMA 的源地址与目的地址设为相同的地址；字节长度设置为期望覆盖的内存块。地址访问必须是 128-bit 对齐，并且整块的内存范围长度必须是 128 bits 的整数倍。在 IDMA 从 LL2 读取数据时，对于存在有效校验信息的 128-bit 字，EDC 硬件会纠正可能存在于其中的 1-bit 错误。当 IDMA 把数据回写到相同的地址时，EDC 会对数据产生校验信息并标识其为有效。

刷新操作通常是在 1-bit 错误中断服务函数中进行。但是在 1-bit 错误发生之后 2-bit 错误发生之前，某些数据也许不会被访问，在没有访问时 1-bit 错误是不会被自动上报的。为了避免这种情况，应该周期性地刷新整块存储器区间来纠正潜在的 1-bit 错误。下面是一段 LL2 EDC 刷新的代码例子。

```
Uint32 uiLL2_scrub_addr=0x800000;
void LL2_EDC_scrub(Uint32 uiByteCnt)
{
  Uint32  uiLL2EndAddress= 0x00880000;
  TDSP_Board_Type DSP_Board_Type;

  uiByteCnt &= 0xFFFFFFF0;  //size must be multiple of 128 bits

  IDMA_copy(uiLL2_scrub_addr, uiLL2_scrub_addr, uiByteCnt, DMA_NO_WAIT);

  uiLL2_scrub_addr+= uiByteCnt;

  DSP_Board_Type = KeyStone_Get_dsp_board_type();
  if((DUAL_NYQUIST_EVM == DSP_Board_Type)
    ||(C6670_EVM == DSP_Board_Type)
    ||(TCI6614_EVM == DSP_Board_Type))
  {
    uiLL2EndAddress= 0x00900000;
  }

  //wrap back
  if(uiLL2_scrub_addr >= uiLL2EndAddress)
    uiLL2_scrub_addr=0x800000;
}
```

通常，这个函数可以在一个定时中断中调用。如在一个 600 秒周期的定时中断中调用该函数。

```
      /*scrub 1024 bytes in LL2 for EDC*/
      LL2_EDC_scrub(1024);
```

这样， 1MB 的存储区间会每 7 天被刷新一遍。

由于刷新操作会与正常的内存操作相竞争，因此会影响正常内存操作的性能。所以刷新操作不能太频繁，但是必须在 2-bit 错误产生前完成。在设计时必须权衡考虑。

LL2 EDC 功能验证：通过设置 L2EDCMD 寄存器中的 SUSP 比特可以暂停 LL2 EDC 逻辑。使用该特性，可以软件模仿 EDC 错误并验证 EDC 功能。与本文对应的例程中提供了验证 LL2 EDC 功能的代码，对应函数 LL2_ED_test()。

3.3 SL2 错误检测与纠正

对共享存储器 SL2 的基本信息，参考“KeyStone Architecture Multicore Shared Memory Controller User Guide（SPRUGW7）”。

校验比特产生与核对：有两种机制用于 MSMC 校验信息的产生与检测：

1, 对任意 master 发起的 256-bit 内存段的写操作时，校验信息会被更新并设置为有效。小于 256 bits 的写操作会使校验信息失效。当 DSP master 发起 256-bit 内存段的的读操作时，校验信息会被检查。

2, MSMC 包含一个后台错误纠正硬件称作刷新引擎，用于周期刷新存储器的内容。刷新的周期数可以通过 SMEDCC 寄存器中的 REFDEL 比特域来配置，每次刷新会读取并回写大小是 4 个 32 字节的块。在检测并纠正 1-bit 或者检查到 2-bit 错误时，刷新引擎还会上报 EDC 错误。在 MSMC 用户手册中有具体的机制细节描述。

DSP 复位后，MSMC 硬件会使校验信息失效，并重新初始化校验信息。在第一次读 MSMC 存储器时，软件必须先检查 SMEDCC 中的 PRR 比特（校验 RAM 是否准备好的状态信息）。

错误检测与纠正配置：DSP 复位后 SL2 EDC 逻辑的刷新引擎被使能，并且会在后台产生校验信息。软件不需要像 LL2 EDC 一样使用 DMA 进行存储器刷新，只需要查询 SMEDCC 寄存器中的 PRR(校验 RAM 准备)比特位来确认校验比特已经产生。为了使能错误纠正，SMEDCC 中的 ECM 比特同样应该使能。请注意，错误纠正逻辑会对从 SL2 的读操作增加 1 cycle 的时延（访问流水线增加了一级），不过访问吞吐量并不会降低。

下面是使能 MSMC EDC 功能的例程：

```
void KeyStone_SL2_EDC_enable(Uint32 scrubCnt)
{
  if(gpMSMC_regs->CFGLCKSTAT&CSL_MSMC_CFGLCKSTAT_WSTAT_MASK)
    CSL_MSMC_unlockNonMPAX();

  /*Software must wait for the PRR (Parity RAM Ready) bit before making
  the first read access to MSMC RAM after reset.*/
  while(0==(gpMSMC_regs->SMEDCC&CSL_MSMC_SMEDCC_PRR_MASK));

  /* set scrubbing period value */
  if(scrubCnt>255)
    scrubCnt= 255;
  CSL_MSMC_setCounterBankRefreshRead(scrubCnt); //the scrubbing engine works every
scrubCnt*1024 cycle*/

  /* clear EDC errors and enable EDC event*/
  gpMSMC_regs->SMIRC = 0xf;
  gpMSMC_regs->SMIESTAT |= (CSL_MSMC_SMIESTAT_NCSIE_MASK
                | CSL_MSMC_SMIESTAT_CSIE_MASK
                | CSL_MSMC_SMIESTAT_NCEIE_MASK
                | CSL_MSMC_SMIESTAT_CEIE_MASK);

  //enable SL2 EDC
  CSL_MSMC_setECM(1);

  CSL_MSMC_lockNonMPAX();
}
```

错误上报机制：MSMC 用户手册中有详细的错误上报机制信息，这里总结如下表。

表 5 MSMC EDC 1-bit 错误上报

requester	Scrubbing engine access	None Scrubbing engine access
Status register (SMESTAT)	“CSES” field = 1	“CEES” field = 1
Error address	Error address is recorded by “SMCEA” register	Error address is recorded by “SMCERRAR” and “SMCERRXR” register

Correctable error counter	“SCEC” field of SMSECC register is increased by 1	No counter register
Non-correctable error counter	No update	No counter register

请注意，由刷新引擎上报的错误地址是从 0 开始的地址偏移，而为非刷新访问记录的错误地址是器件中从 0x0C000000 开始的 SL2 地址。

表 6 MSMC EDC 2-bit 错误上报

requester	Scrubbing engine access	None Scrubbing engine access
Status register (SMESTAT)	“NCSES” field = 1	N“CEES” field = 1
Error address	Error address is recorded by “SMNCEA”	Error address is recorded by “SMNCERRAR” and “SMNCERRXR” register
Correctable error counter	No update	No counter register
Non-correctable error counter	“SNCEC” field of SMSECC register is increased by 1	No counter register

MSMC EDC 功能验证：可以通过设置 SMEDCTST 寄存器中的 PFn 比特位（bit0~3）来暂停 MSMC EDC 逻辑。SMEDCTST 的地址偏移是 0x58。每个 SL2 RAM bank 对应 PFn 中一个比特（PF0~3 与 bank0~3 依次对应），每个比特可以用于禁止对校验 RAM 的写操作。这样可以冻结 bank 对应的校验 RAM，因此可以通过故意注入错误来破坏 SL2 存储内容与校验信息的一致性，从而测试检测纠正逻辑。具体的顺序如下：

1 向测试 bank 中的某一个位置写一个已知值，这样可以正确地为这个位置初始化一个校验值。

2 向 SMEDCSTST 对应的 PF 比特写 1 以冻结该校验值。

3 向上述被写的位置写任意字节来改变该位置的数值，如果检验纠正功能则写一个 1-bit 差异的值，如果检验检测功能则写一个存在 2-bit 差异的值。此时该位置的校验值与其存储的数值没有同步。

4 读回该位置的值，将会产生所选类型的校验错误。

与本文对应的例程中提供了相应的代码用于验证 SL2 EDC 功能，对应的函数为 SL2_EDC_test()。

4 其它鲁棒性特性

4.1 看门狗定时器

对应看门狗定时器的基本知识，请参考“KeyStone Architecture Timer64 User Guide（SPRUGV5）”中“看门狗定时器模式”章节。

定时器 0~（N-1）可用于 N 个 core 的看门狗。在 TCI6614 中定时器 8 是 ARM 的看门狗定时器。

在看门狗模式下，定时器倒计时到 0 时产生一个事件。需要由软件在倒计时终止前向定时器写数，然后计数重新开始。如果计数到 0，会产生一个定时器事件。看门狗定时器事件可以触发本核复位、器件复位或者 NMI 异常，这可以通过配置相应器件手册中描述的“复位复用寄存器（RSTMUXx）”来选择。

使看门狗事件触发 NMI 异常具有更高的灵活性，在 NMI 异常服务函数中，错误的原因及某些关键的状态信息可以被记录下来，或者上报给上位机来进行故障分析，然后如果它不能自恢复则可以再由软件来复位器件。

4.2 EDMA 错误检测

关于基本的 EDMA CC 错误信息可以参考“KeyStone Architecture Enhanced Direct Memory Access(EDMA3)Controller User Guide（SPRUGS5）”中的“错误中断”章节。

关于基本的 EDMA TC 错误信息可以参考“KeyStone Architecture Enhanced Direct Memory Access(EDMA3) Controller User Guide（SPRUGS5）”中的“错误产生”章节。

所有的 EDMA 错误事件可作为异常被路由到 CorePac。

事件丢失错误是一种最常见的 EDMA CC 错误，意味着 EDMA 不能按要求及时完成数据的传输，或者错误的事件触发了不应该的 EDMA 传输。

总线错误是一种最常见的 EDMA TC 错误，通常意味着 EDMA 访问了错误的地址（如预留地址或受保护的地址）。

4.3 中断丢失检测

中断丢失或遗漏是实时系统中常见也是常被忽略的问题。中断丢失检测是一种用于捕捉这种异常的有效方法。对基本的中断丢失检测信息参考“TMS320C66x DSP CorePac User Guide（SPRUGW0）”中“中断错误事件”章节。

软件系统应该对路由到 DSP core 且有对应软件服务的中断使能中断丢失检测。在所有中断配置完毕后可以添加如下代码使能中断丢失检测:

```
gpCGEM_regs->INTDMASK= ~IER; /*only monitor drop of enabled interrupts*/
```

注意，当使能中断丢失检测并在 CCS/Emulator 下使用断点或单步进行调测时，由于在仿真停止时中断没有被响应，所有此时中断丢失错误上报的概率很高。如果想忽略它，可以在调测时暂时对某些或全部中断关闭中断丢失检测，但是注意不要忘记在正式发布的程序中重新使能该功能。

5 异常处理

关于异常处理的基本信息参考“TMS320C66x DSP CPU and Instruction Set Reference Guide（SPRUGH7）”中“CPU 异常”一节。

关于中断或异常事件路由的基本信息参考“TMS320C66x DSP CorePac User Guide（SPRUGW0）”中“中断控制器”章节。

5.1 异常事件路由

所有源自或由 CorePac 触发的错误事件均直接路由到 CorePac 的中断控制器。常被当作异常处理的错误如下表所示。

表 7 直接送往 CorePac INTC 的异常事件

Event Number	Event	Description
10	MSMC_mpf_error_n	本核对应的 Privilege ID 产生的 MSMC 存储控制器保护错误。如本核编程的 EDMA 触发了 MSMC 内存保护。
96	INTERR	被丢弃的 CPU 中断事件
97	EMC_IDMAERR	非法的 IDMA 参数
110	MDMAERREVT	XMC VBUSM 错误事件，可能由于 MPAX 保护或来自终端的总线错误的违法性导致。
113	L1P_ED	DMA 访问 L1P 时检测到的单一比特错误。 DSP core 执行代码时的单一比特 L1P 错误将会触发内部指令获取异常。
117	LL2_ED2	LL2 中检测到的不可纠正错误
119	SYS_CMPA	本地配置 INTC 及省电控制时产生的内存保护故障
120	L1P_CMPA	DSP core 访问 L1P 时产生的内存保护故障
121	L1P_DMPA	DMA 访问 L1P 时产生的内存保护故障
122	L1D_CMPA	DSP core 访问 L1D 以及其他内存读取最终经过 L1D 控制器的操作产生的内存保护故障
123	L1D_DMPA	DMA 访问 L1D 产生的内存保护故障
124	LL2_CMPA	DSP core 访问 L2 产生的内存保护故障
125	LL2_DMPA	DMA 访问 L2 产生的内存保护故障
126	EMC_CMPA	本地配置区域 0x01000000 – 0x01BFFFFF 产生的内存保护故障
127	EMC_BUSERR	全局配置区域 0x01C00000 – 0x07FFFFFF 产生的总线错误中断

一些其他非致命的错误事件，如可纠正的 LL2 EDC 错误，应该被路由到中断而非异常。

源自或者由器件中共享模块触发的错误事件被路由到 CIC。CIC 基本信息参考“KeyStone Architecture Chip Interrupt Controller(CIC) User Guide（SPRUGW4）”。

CIC 事件中常被当作异常处理的事件如下表所示。

表 8 通过 CIC 路由的异常事件

Event Number	Event	Description
0	EDMACC_ERRINT	EDMA3CC1 错误中断
2,3,4,5	EDMATC_ERRINT	EDMA3CC1 EDMA3TC0,1,2,3 错误中断
16	EDMACC_ERRINT	EDMA3CC2 错误中断
18,19,20,21	EDMATC_ERRINT	EDMA3CC2 EDMA3TC0,1,2,3 错误中断
32	EDMACC_ERRINT	EDMA3CC3 错误中断
34,35	EDMATC_ERRINT	EDMA3CC3 EDMA3TC0,1 错误中断
99	MSMC_dedc_nc_error	MSMC SRAM 读操作检测到的不可纠正软错误
100	MSMC_scrub_nc_error	周期刷新 MSMC 时检测到的不可纠正软错误
102~109	MSMC_mpf_error	PrivID 8~15 对应的系统 master 触发的内存保护错误
170~173	MSMC_mpf_error	对 4 coreDSP（TCI6614，C6670 …）中相应 PrivID4~7 的系统 master 触发的内存保护错误
90, 92, 94, 96	MPU_INTD	**C6678, C6670** 上 MPU0~3 产生的非法访问中断
174, 180	MPU_INTD	**C6670 上 MPU4,5 产生的非法访问中断**
23	MPU_Combined_Address_Error	TCI6614 上 MPU0~7 寄存器空间中保留寄存器被非法访问
37	MPU_Combined_PROT_Error	TCI6614 上 MPU0~7 保护非法中断

每种这样的异常事件只能路由到一个 CorePac。通常所有的这些事件被路由到一个 CorePac。

下图描述 DSP core 内部控制异常处理的开关。

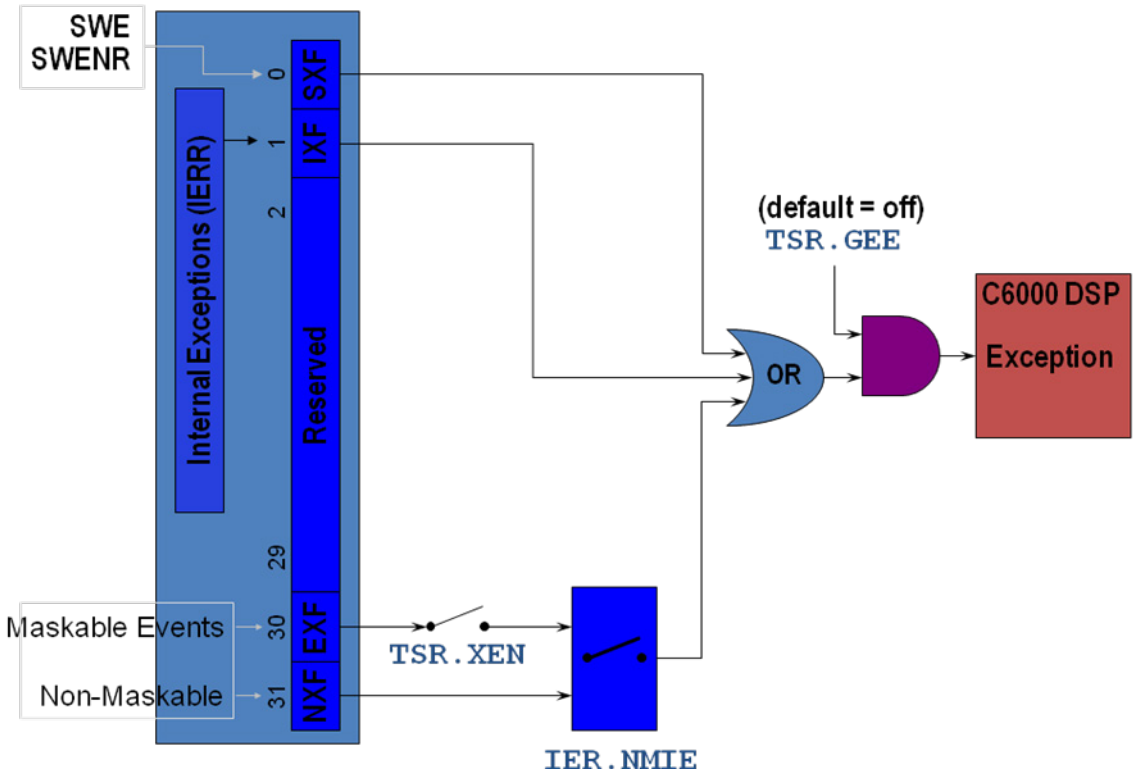

图 2 DSP 核异常控制开关

一旦软件置位 TSR.GEE 及 IER.NMIE，不能再由软件清除，只能在复位后被清除。

TSR.XEN 可以由软件置位并清除。XEN 可以在进入异常服务函数中由硬件自动清除，并在退出异常服务函数时自动恢复原来的状态。

因此，默认情况下，在中断服务函数中，TSR.GEE=1，IER.NMIE=1 及 TSR.XEN=0.

5.2 异常服务函数

异常函数中应该记录或上报异常原因及相关信息，用于故障分析。

关键的记录信息是 NRP。NRP 是异常返回指针，通常用于确定异常触发的位置。

实际上，非法操作与 NRP 捕获之间的时延大概在 10~100 个 DSP Core cycles 之间，具体的时延取决于很多因素，如操作类型，产生异常事件的模块等等。例如对于向一个被 MPU 保护的寄存器执行写操作，其时延包括：从 DSP core 到寄存器的写指令时延；错误事件从 MPU 到 CIC 然后到 CorePac 异常模块的路由时延。因此，当我们获得 NRP 后，应从 NRP 指向的位置向后搜索大概 10~100cycles 来找有问题的操作。

不过，某些异常 NRP 是没有意义的，例如，对于指令获取异常及非法操作码异常。这通常发生在当程序跳转到一个非法的地址时，这时 NRP 也指向一个非法的地址。我们真正想知道的是在程序跳转到非法地址前到底发生了什么，但是这并不能从 NRP 推导出来。在这种情况下，寄存器 B3，A4，B4，B14 及 B15 也许会有所帮助。B3 可能还保存着上次函数调用的返回指针；A4 及 B4 也许保存着上次函数调用的参数；B15 是栈指针；B14 是指向某些全局变量的数据指针。更多的细节可以参考“TMS320C6000 Optimizing Compiler User Guide(SPRUG187)” 中“7.4 函数结构及调用约定”章节。根据这些信息，我们也许可以推导出在程序跳转到非法地址前发生了什么。注意，B3，A4，B4 可能在异常发生前已经被修改用于保存其它信息，所以它们也许不是有用的。实际上，B3，A4，B4 包含有价值信息的概率还是很高的，所以这些寄存器是值得记录并分析的。

通用寄存器的值不能用 C 代码记录，而必须用汇编代码来记录。下面的例子是将 B3,A4,B4,B14,B15 寄存器记录在“exception_record”中，然后调用“Exception_service_routine”。

```
       .ref Exception_service_routine
       .ref exception_record
;---------------------------------------------------------------
       .sect ".text"
;interrupt vector for NMI
NMI_ISR:
       STW         B1,*-B15[1]

       ;save some key registers when exception happens
       MVKL  exception_record,B1
       MVKH  exception_record,B1

       STW         B3, *+B1[0]
       STW         A4, *+B1[1]
       STW         B4, *+B1[2]
       STW         B14, *+B1[3]
       STW         B15, *+B1[4]

       ;jump to exception service routine
       MVKL        Exception_service_routine, B1
       MVKH        Exception_service_routine, B1
       B           B1

       LDW         *-B15[1],B1
       NOP         4
```

其它需要记录的基本信息有：EFR,IERR,NTSR,TSCL/TSCH. EFR 用于判决异常类型：内部、外部或是 NMI。对于内部异常，内部异常的原因记录在 IERR。NTSR 记录异常发生时的 DSP core 状态。记录的 TSCL/TSCH 用于确定异常发生前器件运行的时长。

对于外部异常，通过检查 INTC 及 CIC 标志寄存器来决定异常原因。对应一个特定的异常，往往有特定的状态寄存器可以检查、记录或上报。例如对应内存保护异常，需要记录的关键信息是故障地址。参考各模块的用户指南了解相关状态或标志的更多细节。

通常，异常服务函数将这些异常信息保存在一个类似如下的数据结构中。

```
typedef  struct{
```

```
  volatile Uint32 MEXPFLAG[4];  /*copy of the MEXPFLAG0..3 registers */
  volatile Uint32 CIC_STATUS[6];  /*copy of the CIC status registers */
  Exception_Info info;
} External_Exception_Status;

typedef  union  {
  volatile Uint32 IERR; /*copy of the IERR register */
  External_Exception_Status ext_sts;
} Exception_Status;

typedef struct {
  volatile Uint32 B3; /*copy of B3 register (return pointer of caller) */
  volatile Uint32 A4; /*copy of the A4 register (first input parameter of caller)*/
  volatile Uint32 B4; /*copy of the B4 register (second input parameter of caller)*/
  volatile Uint32 B14; /*copy of the B14 register (data pointer)*/
  volatile Uint32 B15; /*copy of the B15 register (stack pointer)*/
  volatile Uint32 TSCL; /*copy of the TSCL register (time stamp)*/
  volatile Uint32 TSCH; /*copy of the TSCH register (time stamp)*/
  volatile Uint32 NTSR; /*copy of the NTSR register */
  volatile Uint32 NRP; /*copy of the NRP register */
  volatile Uint32 EFR; /*copy of the EFR register */
  Exception_Status status;
} Exception_Record;
```

可以在异常服务函数中将这些数据结构中的信息传递给主机，或者将其导出来进行错误分析。

通常异常服务函数处理的错误是致命的，用户不应该期望从异常服务函数中返回。另外，软件也不总是能从异常服务函数中安全返回，阻止从异常中安全返回的条件有：

1, 被异常终止的 SPLOOPs 不能正确地重新开始。在返回前应该核实 NTSR 中的 SPLX 比特数值为 0.

2, 中断被堵塞时发生的异常不能正确地重新开始。在返回前应该核实 NTSR 中的 IB 比特数值为 0.

3, 在不能被安全中断的代码处（如一个保护多个赋值的紧凑循环）发生的异常不能正确地返回。编译器通常会在代码中的这些地方关闭中断；查看 NTSR 中的 GIE 比特值为 1 来验证满足这个条件。

4, NRP 不是一个合法的地址。

所以通常异常服务函数以一个 while(1)循环作为结束。

默认情况下在异常服务程序中，TSR.GEE=1,IER.NMIE=1 及 TSR.XEN=0.即在异常服务程序中 NMI 及内部异常是使能的。

当一个使能的异常发生在第一个异常服务程序中时，复位向量指向的程序会被执行。这时 NTSR 和 NRP 不会发生改变。TSR 复制到 ITSR，此时的 PC 复制到 IRP。此时为了避免其他外部异常，硬件将 TSR 设置为默认的异常处理值，NMIE 中的 IER 比特被清零。

通常中断服务表中的复位向量是跳转到程序起始位置如_c_int00，这样，嵌套异常会重启程序。然而这并非大部分用户所期望的，我们通常期望的是异常发生时在异常服务程序执行完后结束程序。为了避免嵌套异常导致程序重启，可以给嵌套异常添加一个额外的异常服务程序，用户可以修改复位向量跳转到嵌套异常服务程序。在 KeyStone 器件中，加载程序不依赖于复位向量启动程序，所以修改复位向量不会影响程序的加载。

6 例程

本文相关的例程可以在 TCI6614 EVM, C6670 EVM 及 C6678 EVM 上跑通。例程可以从以下链接下载：

http://www.deyisupport.com/question_answer/dsp_arm/c6000_multicore/f/53/t/47664.aspx

如下为工程目录结构：

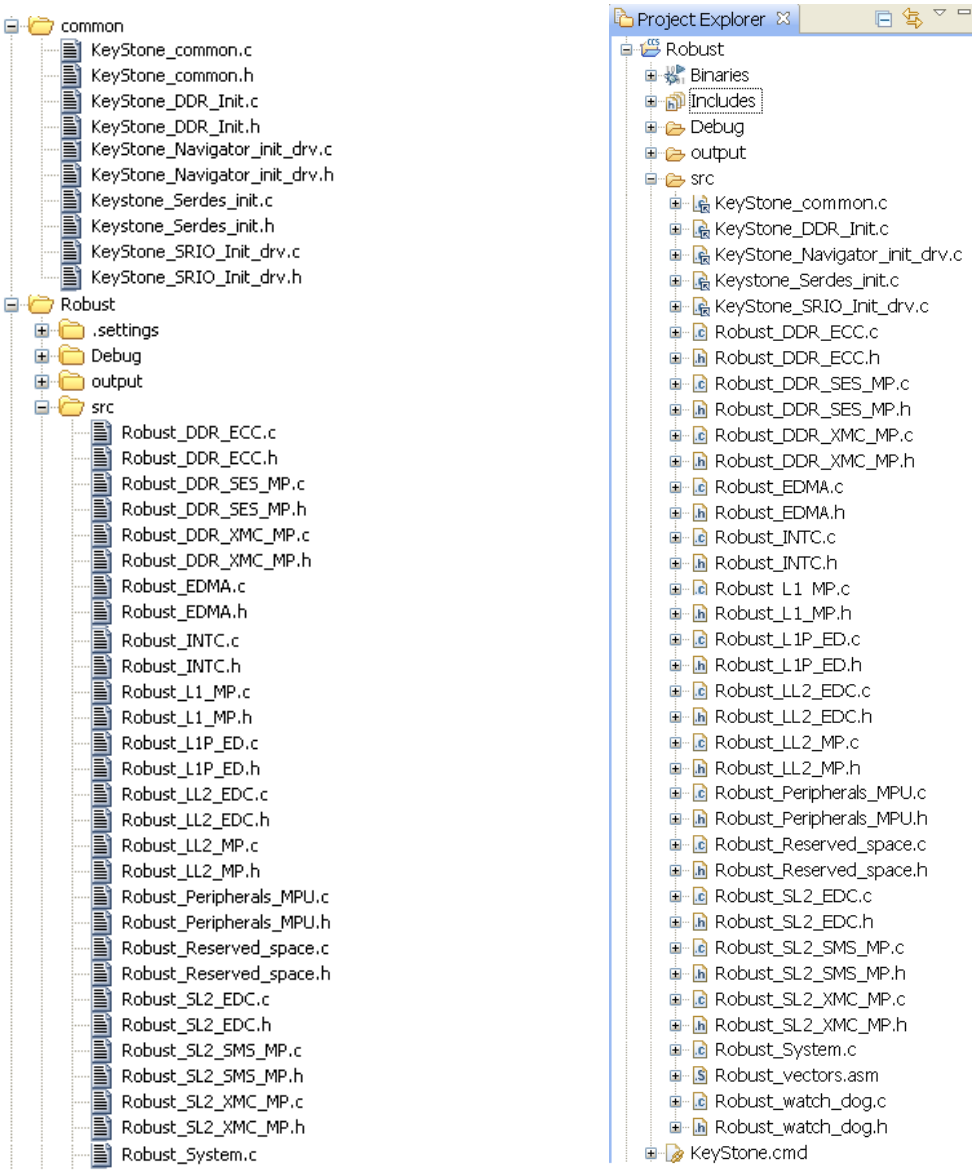

图 3 例程目录结果

“common“文件夹中包含通用代码如 DDR 初始化及 DMA、定时器、多核导航器、SRIO 驱动等。内存保护初始化代码、EDC 及异常处理的代码包含在 KeyStone_common.c.

“src”文件夹中的每个 c 文件包含一个测试用例代码。主函数在 “Robust_System.c”. 在“Robust_System.c “的开头有一些宏开关，每个开关用于使能或关闭一个测试用例。

```
#define L1D_MP_TEST                1
#define L1P_MP_TEST                1
#define LL2_MP_TEST                1
#define XMC_SL2_MP_TEST            1
#define XMC_DDR_MP_TEST            1
#define SMS_SL2_MP_TEST            1
#define SES_DDR_MP_TEST            1
#define PERIPHERAL_MP_TEST         1
#define L1P_ED_TEST                1
#define LL2_EDC_TEST               1
#define SL2_EDC_TEST               1
#define EDMA_ERROR_TEST            1
#define WATCH_DOG_TEST             1
#define RESERVED_SPACE_TEST        1
#define INT_DROP_TEST              1
```

如果使能多个测试用例，每个用例会依次执行。由于程序并不能总是安全地从异常服务程序中返回，因此有可能在一个测试用例后输出如下信息，然后测试流程被终止。

```
Exception happened at a place can not safely return!
```

如果出现这种情况，可以关闭这个测试用例然后重新测试其他的用例。

在 EVM 上运行例程的步骤如下：

1, 解压例程， 将 CCS workspace 切换到解压后的文件夹；

2, 在 workspace 中导入工程；

3, 如果发生代码修改对工程重新编译，也许需要在编译选项中修改 CSL 保护路径；

4, 设置 EVM 板上的器件加载模式为 No boot 模式；

5, 将代码加载到 DSP core0，运行；

6, 查看 CCS stdout 窗口浏览测试结果。

如下为 TCI6614 上的测试结果。

```
Initialize DSP main clock = 122.88MHzx236/29 = 999MHz
Initialize DDR speed = 66.667x20/1= 1333.3
SRIO path configuration 4xLaneABCD
Enable Exception handling...
====================-L1D memory protection test-==================
!!!DMA write to L1D at 0x00F00000...
External exception happened. MEXPFLAG[0]=0x800000.
External exception happened. MEXPFLAG[3]=0x8000000.
  EDMA module 0 error
    TC 0 error happened
```

```
      EDMA3TC has detected an error at source or destination address.
      write error (10). TCC= 0. TCINTEN= 1. TCCHEN= 0
  Event 123: DMC_DMPA DMA memory protection fault for L1D
  memory protection exception caused by master with ID 0 at 0xf00000
  Supervisor Write violation
NRP=0xc0124b8, NTSR=0x1000d, TSCH= 0x0, TSCL= 0xc594e5
 B3=0xc012468, A4=0x0, B4= 0x1, B14= 0x839fc0, B15= 0x838fb0

!!!DSP core write to L1D at 0x00F00100...
External exception happened. MEXPFLAG[3]=0x4000000.
  Event 122: DMC_CMPA CPU memory protection fault for L1D (and other memory read
finally goes through the L1D controller)
  memory protection exception caused by local access at 0xf00100
  Supervisor Write violation
NRP=0xc0124fa, NTSR=0x1000d, TSCH= 0x0, TSCL= 0xc6ff1d
 B3=0xc0124e0, A4=0x2b, B4= 0xbfbfbfbf, B14= 0x839fc0, B15= 0x838fb0

!!!DSP core write to CGEM_regs->L1DMPPA[0] register at 0x184ae40 without unlocking...
External exception happened. MEXPFLAG[3]=0x4000000.
  Event 122: DMC_CMPA CPU memory protection fault for L1D (and other memory read
finally goes through the L1D controller)
  memory protection exception caused by local access at 0x184ae40
  Supervisor Write violation
NRP=0xc012548, NTSR=0x1000d, TSCH= 0x0, TSCL= 0xc80c2a
 B3=0xc012528, A4=0x57, B4= 0x184ae40, B14= 0x839fc0, B15= 0x838fb0
====================-L1P memory protection test-==================
!!!DMA write to L1D at 0x00E00000...
External exception happened. MEXPFLAG[0]=0x800000.
External exception happened. MEXPFLAG[3]=0x2000000.
  EDMA module 0 error
    TC 0 error happened
      EDMA3TC has detected an error at source or destination address.
      write error (10). TCC= 0. TCINTEN= 1. TCCHEN= 0
  Event 121: PMC_DMPA DMA memory protection fault for L1P
  memory protection exception caused by master with ID 0 at 0xe00000
  Supervisor Write violation
NRP=0xc0123b8, NTSR=0x1000d, TSCH= 0x0, TSCL= 0xc8e9a2
 B3=0xc012368, A4=0x0, B4= 0x1, B14= 0x839fc0, B15= 0x838fb0

!!!DSP core write to CGEM_regs->L1PMPPA[0] register at 0x184a640 without unlocking...
External exception happened. MEXPFLAG[3]=0x1000000.
  Event 120: PMC_CMPA CPU memory protection fault for L1P
  memory protection exception caused by local access at 0x184a640
  Supervisor Write violation
NRP=0xc01240c, NTSR=0x1000d, TSCH= 0x0, TSCL= 0xca8754
 B3=0xc0123f0, A4=0x57, B4= 0x1800000, B14= 0x839fc0, B15= 0x838fb0
====================-LL2 memory protection test-==================
EDMA write to 0x10828000, which is allowed for external access...

!!!EDMA write to 0x10820000, which is protected against external access...
External exception happened. MEXPFLAG[0]=0x800000.
External exception happened. MEXPFLAG[3]=0x20000000.
  EDMA module 0 error
    TC 0 error happened
      EDMA3TC has detected an error at source or destination address.
      write error (10). TCC= 0. TCINTEN= 1. TCCHEN= 0
  Event 125: UMC_DMPA DMA memory protection fault for L2
  memory protection exception caused by master with ID 0 at 0x820000
  Supervisor Write violation
NRP=0xc010dee, NTSR=0x1000d, TSCH= 0x0, TSCL= 0xd89b30
 B3=0xc010d9a, A4=0x0, B4= 0x1, B14= 0x839fc0, B15= 0x838fb0
```

```
!!!DSP core write to CGEM_regs->L2MPPA[0] register at 0x184a200 without unlocking...
External exception happened. MEXPFLAG[3]=0x10000000.
  Event 124: UMC_CMPA CPU memory protection fault for L2
  memory protection exception caused by local access at 0x184a200
  Supervisor Write violation
NRP=0xc010e48, NTSR=0x1000d, TSCH= 0x0, TSCL= 0xda387e
 B3=0xc010e28, A4=0x56, B4= 0xffffffff, B14= 0x839fc0, B15= 0x838fb0
====================-XMC SL2 memory protection test-==============
DSP core write to 0xc110400 which is write-only...
DSP core read from 0xc12c808 which is read-only...

!!!DSP core read from 0xc11883f which is write-only...
External exception happened. MEXPFLAG[3]=0x4000000.
  Event 122: DMC_CMPA CPU memory protection fault for L1D (and other memory read
finally goes through the L1D controller)
  memory protection exception caused by local access at 0xc11883f
  Supervisor Read violation
NRP=0xc01204c, NTSR=0x1000d, TSCH= 0x0, TSCL= 0xeaa097
 B3=0xc012034, A4=0x38, B4= 0x20, B14= 0x839fc0, B15= 0x838fb0

!!!DSP core write to 0xc12d44c which is read-only...
External exception happened. MEXPFLAG[3]=0x4000.
  Event 110: MDMAERREVT XMC VBUSM error event
  memory protection exception caused by local access at 0xc12d440
  Supervisor Write violation
NRP=0xc01208a, NTSR=0x1000d, TSCH= 0x0, TSCL= 0xeb9614
 B3=0xc012070, A4=0x36, B4= 0xc12d44c, B14= 0x839fc0, B15= 0x838fb0

!!!DSP core write to XMC_regs->XMPAX[0].XMPAXL register at 0x8000000 without
unlocking...
External exception happened. MEXPFLAG[3]=0x4000.
  Event 110: MDMAERREVT XMC VBUSM error event
  memory protection exception caused by local access at 0x8000000
  Supervisor Write violation
NRP=0xc0120c8, NTSR=0x1000d, TSCH= 0x0, TSCL= 0xec9fcf
 B3=0xc0120ac, A4=0x5b, B4= 0x8000000, B14= 0x839fc0, B15= 0x838fb0
====================-XMC DDR memory protection test-==============
DSP core write to 0x81000400 which is write-only...
DSP core read from 0x81020808 which is read-only...

!!!DSP core read from 0x8100883f which is write-only...
External exception happened. MEXPFLAG[3]=0x4000000.
  Event 122: DMC_CMPA CPU memory protection fault for L1D (and other memory read
finally goes through the L1D controller)
  memory protection exception caused by local access at 0x8100883f
  Supervisor Read violation
NRP=0xc0122b4, NTSR=0x1000d, TSCH= 0x0, TSCL= 0xfd0700
 B3=0xc0122a0, A4=0x39, B4= 0x20, B14= 0x839fc0, B15= 0x838fb0

!!!DSP core write to 0x8102144c which is read-only...
External exception happened. MEXPFLAG[3]=0x4000.
  Event 110: MDMAERREVT XMC VBUSM error event
  memory protection exception caused by local access at 0x81021440
  Supervisor Write violation
NRP=0xc0122f2, NTSR=0x1000d, TSCH= 0x0, TSCL= 0xfdfcf9
 B3=0xc0122d8, A4=0x37, B4= 0x8102144c, B14= 0x839fc0, B15= 0x838fb0
====================-SMS SL2 memory protection test-==============
EDMA copy from 0xc128000 (read-only) to 0xc100000 (write-only)...

!!!EDMA copy from 0xc108000 (write-only) to 0xc129000 (read-only)...
```

```
External exception happened. MEXPFLAG[0]=0x800400.
  EDMA module 0 error
    TC 0 error happened
      EDMA3TC has detected an error at source or destination address.
      read error (2). TCC= 0. TCINTEN= 1. TCCHEN= 0
  Event 10 : MSMC_mpf_error_n MSMC Memory protection fault indicators for local CorePac
  memory protection exception caused by master 16 (PrivID= 0) at address 0xc108000
NRP=0xc00cc8c, NTSR=0x1000d, TSCH= 0x0, TSCL= 0x10df64d
 B3=0xc00cc66, A4=0x0, B4= 0x1, B14= 0x839fc0, B15= 0x838f68

!!!DSP core write to msmcRegs->SMS_MPAX_PER_PRIVID[0].SMS[0].MPAXL register at
0xbc00200 without unlocking...
External exception happened. MEXPFLAG[3]=0x4000.
  Event 110: MDMAERREVT XMC VBUSM error event
  MDMA write status error detected
  XID (Transaction ID)= 8
  Privilege error
NRP=0xc00cd02, NTSR=0x1000d, TSCH= 0x0, TSCL= 0x10f889d
 B3=0xc00ccd4, A4=0x6f, B4= 0xbc00200, B14= 0x839fc0, B15= 0x838f68
--------------------SRIO SMS SL2 memory protection test---------------
SRIO copy from 0xc12a000 (read-only) to 0xc108000 (write-only)...

!!!SRIO copy from 0xc12a000 (read-only) to 0xc129000 (read-only)...
External exception happened. MEXPFLAG[0]=0x800000.
  memory protection exception caused by master 54 (PrivID= 9) at address 0xc129000
NRP=0xc00cfc6, NTSR=0x1000d, TSCH= 0x0, TSCL= 0x110be59
 B3=0xc00cf92, A4=0xc12a020, B4= 0x3, B14= 0x839fc0, B15= 0x838f68
====================-SES DDR memory protection test-==============
EDMA copy from 0x81024000 (read-only) to 0x81010000 (write-only)...

!!!EDMA copy from 0x81018000 (write-only) to 0x81025000 (read-only)...
External exception happened. MEXPFLAG[0]=0x800400.
  EDMA module 0 error
    TC 0 error happened
      EDMA3TC has detected an error at source or destination address.
      read error (1). TCC= 0. TCINTEN= 1. TCCHEN= 0
  Event 10 : MSMC_mpf_error_n MSMC Memory protection fault indicators for local CorePac
  memory protection exception caused by master 16 (PrivID= 0) at address 0x81018000
NRP=0xc00c54c, NTSR=0x1000d, TSCH= 0x0, TSCL= 0x120c9d0
 B3=0xc00c526, A4=0x0, B4= 0x1, B14= 0x839fc0, B15= 0x838f68

!!!DSP core write to msmcRegs->SES_MPAX_PER_PRIVID[0].SES[0].MPAXL register at
0xbc00600 without unlocking...
External exception happened. MEXPFLAG[3]=0x4000.
  Event 110: MDMAERREVT XMC VBUSM error event
  MDMA write status error detected
  XID (Transaction ID)= 12
  Privilege error
NRP=0xc00c5c2, NTSR=0x1000d, TSCH= 0x0, TSCL= 0x1225bbf
 B3=0xc00c594, A4=0x6f, B4= 0xbc00600, B14= 0x839fc0, B15= 0x838f68
--------------------SRIO SES DDR3 memory protection test---------------
SRIO copy from 0x81026000 (read-only) to 0x81018000 (write-only)...

!!!SRIO copy from 0x81026000 (read-only) to 0x81025000 (read-only)...
External exception happened. MEXPFLAG[0]=0x800000.
  memory protection exception caused by master 54 (PrivID= 9) at address 0x81025000
NRP=0xc00c874, NTSR=0x1000d, TSCH= 0x0, TSCL= 0x1239329
 B3=0xc00c852, A4=0x81026020, B4= 0x4, B14= 0x839fc0, B15= 0x838f68
====================-peripherals protection test-=================
!!!Write to PLL register at 0x02310110 which is protected by MPU0...
External exception happened. MEXPFLAG[0]=0x800000.
```

```
External exception happened. MEXPFLAG[3]=0x80000000.
  MPU 0 protection violation error
  The MSTID 8 (PRIVID 0) triggered MPU error at 0x2310110 with secure access
  Supervisor write fault!
  Event 127: EMC_BUSERR Bus Error Interrupt for global configuration space between
0x01C00000 - 0x07FFFFFF
  CFG write status error detected
  XID (Transaction ID)= 13
  Privilege error
NRP=0xc013664, NTSR=0x1000d, TSCH= 0x0, TSCL= 0x124aaed
 B3=0xc013644, A4=0x45, B4= 0x2310110, B14= 0x839fc0, B15= 0x838fb0
MPU 2 setup 11 of 16 ranges.

DSP core Pop descriptor from queue 2049 at 0x2a2801c in a read-only range...

!!!DSP core Push descriptor to queue 2049 at 0x2a2801c in a read-only range...
External exception happened. MEXPFLAG[0]=0x800000.
External exception happened. MEXPFLAG[3]=0x80000000.
  MPU 2 protection violation error
  The MSTID 8 (PRIVID 0) triggered MPU error at 0x2a2801c with secure access
  Supervisor write fault!
  Event 127: EMC_BUSERR Bus Error Interrupt for global configuration space between
0x01C00000 - 0x07FFFFFF
  CFG write status error detected
  XID (Transaction ID)= 15
  Privilege error
NRP=0xc013592, NTSR=0x1000d, TSCH= 0x0, TSCL= 0x126ad1a
 B3=0xc013564, A4=0x50, B4= 0xc133101, B14= 0x839fc0, B15= 0x838f98

DSP core Push descriptor to queue 2049 at 0x3402801c in a writeable range...
====================L1P ED test==================================
!!!manually generate one bit error in L1P cache for function at 0x800760, and then
execute it...
internal excpetion happened. IERR=0x9.
  Instruction fetch exception
  Opcode exception
  L1P Cache parity check error caused by program fetch at address 0x800760
NRP=0x800742, NTSR=0x1000d, TSCH= 0x0, TSCL= 0x1289879
 B3=0xc012d74, A4=0x1846408, B4= 0x1840024, B14= 0x839fc0, B15= 0x838fa8

!!!manually generate one bit error in L1P cache at 0xe00760, and then read it by DMA...
External exception happened. MEXPFLAG[3]=0x20000.
  Event 113: PMC_ED Single bit error detected during DMA read
  L1P RAM parity check error caused by DMA at address 0xe00760
NRP=0xc012dda, NTSR=0x1000d, TSCH= 0x0, TSCL= 0x129bb2e
 B3=0xc012dd0, A4=0xe00740, B4= 0x40, B14= 0x839fc0, B15= 0x838fa8
====================-LL2 EDC test-================================
--------------------LL2 data EDC test----------------------------
!!!manually generate 3 bit error in data at 0x83a240, and then read it...
External exception happened. MEXPFLAG[3]=0x200000.
  Event 117: UMC_ED2 Uncorrected bit error detected
  LL2 EDC error (non-correctable) at address 0x83a240 caused by L1D access.
  total non-correctable error number= 1, total correctable error number= 0.
NRP=0xc00d9ec, NTSR=0x1000d, TSCH= 0x0, TSCL= 0x12acd22
 B3=0xc00d9d4, A4=0x1846008, B4= 0xcc, B14= 0x839fc0, B15= 0x838fa8

!!!manually generate 2 bit error in data at 0x83a240, and then read it...
External exception happened. MEXPFLAG[3]=0x200000.
  Event 117: UMC_ED2 Uncorrected bit error detected
  LL2 EDC error (non-correctable) at address 0x83a240 caused by L1D access.
  total non-correctable error number= 2, total correctable error number= 0.
```

```
NRP=0xc00da50, NTSR=0x1000d, TSCH= 0x0, TSCL= 0x12c0d5d
 B3=0xc00da38, A4=0x1846008, B4= 0x1, B14= 0x839fc0, B15= 0x838fa8

!!!manually generate one bit error in data at 0x83a240, and then read it...
External exception happened. MEXPFLAG[3]=0x200000.
  Event 117: UMC_ED2 Uncorrected bit error detected
  LL2 EDC error (non-correctable) at address 0x83a240 caused by L1D access.
  total non-correctable error number= 3, total correctable error number= 0.
NRP=0xc00daa8, NTSR=0x1000d, TSCH= 0x0, TSCL= 0x12d4e08
 B3=0xc00da98, A4=0x1846008, B4= 0x6c, B14= 0x839fc0, B15= 0x838fa8

!!!scrub the corrupted data to fix the error...
  LL2 EDC (correctable) at bit 102 of address 0x83a240 caused by DMA access.
  total non-correctable error number= 3, total correctable error number= 1.
IRP= 0xc00dae2, ITSR= 0xd. TSCH= 0x0, TSCL= 0x12e5f6f
read the data again...(no error happens again)
--------------------LL2 code EDC test----------------------------
!!!manually generate one bit error in a function at 0x800660, and then execute the
function...
  LL2 EDC (correctable) at bit 13 of address 0x800660 caused by L1P access.
  total non-correctable error number= 3, total correctable error number= 3.
IRP= 0x800646, ITSR= 0xd. TSCH= 0x0, TSCL= 0x12f588e

one bit error was corrected with previous execution. Execute the function again...(no
error happens again)
====================-SL2 EDC test-================================
!!!manually generate one bit error in a function at 0xc016ec0, and then execute the
function...
SL2 Correctable error occurred at bit 13 of address 0xc016ec0 by PrivID 0 (from C66x
CorePacs)
IRP= 0xc016eb8, ITSR= 0xd. TSCH= 0x0, TSCL= 0x1306b47

one bit error was corrected with previous execution. Execute the function again...(no
error happens again)

!!!manually generate one bit error data at 0xc136000, and then read it...
SL2 Correctable error occurred at bit 8 of address 0xc136000 by PrivID 0 (from C66x
CorePacs)
IRP= 0xc00f7c4, ITSR= 0xd. TSCH= 0x0, TSCL= 0x131358d

!!!manually generate one bit error data at 0xe0000000, and then read it...
SL2 Correctable error occurred at bit 8 of address 0xc136000 by PrivID 0 (from C66x
CorePacs)
IRP= 0xc00f7c6, ITSR= 0xd. TSCH= 0x0, TSCL= 0x131faeb
====================-EDMA Error test-============================
!!!start a NULL transfer with EDMA CC0 Channel 0, to trigger event miss error...
External exception happened. MEXPFLAG[0]=0x800000.
  EDMA module 0 error
    EDMA Channel 0 event missed.
NRP=0xc0133b8, NTSR=0x1000d, TSCH= 0x0, TSCL= 0x1342b69
 B3=0xc01329c, A4=0x1, B4= 0x1, B14= 0x839fc0, B15= 0x838fb0

!!!start a transfer to address 0 with EDMA CC1 Channel 2, to trigger EDMA bus error...
External exception happened. MEXPFLAG[0]=0x800000.
  EDMA module 1 error
    TC 2 error happened
      EDMA3TC has detected an error at source or destination address.
      read error (1). TCC= 2. TCINTEN= 1. TCCHEN= 0
NRP=0xc013460, NTSR=0x1000d, TSCH= 0x0, TSCL= 0x13501ea
 B3=0xc013428, A4=0x0, B4= 0x4, B14= 0x839fc0, B15= 0x838fb0
====================-watch dog timer test-=======================
```

```
start watch-dog timer...
service watch-dog 1 times, at time counter = 4
service watch-dog 2 times, at time counter = 6646748
service watch-dog 3 times, at time counter = 7229707
service watch-dog 4 times, at time counter = 7020074
service watch-dog 5 times, at time counter = 6726619
service watch-dog 6 times, at time counter = 6643649
service watch-dog 7 times, at time counter = 7002163
service watch-dog 8 times, at time counter = 6770232
service watch-dog 9 times, at time counter = 6738152
service watch-dog 10 times, at time counter = 6510560
stop servicing watch-dog, it will timeout and trigger NMI after 3000 ms...
NMI exception happened, normally you should reset the DSP to recover from the problem!
NRP=0xc014108, NTSR=0x1000d, TSCH= 0x0, TSCL= 0xaf397e41
 B3=0xc0140e4, A4=0x4b, B4= 0x1079551, B14= 0x839fc0, B15= 0x838f88
==================-Reserved space access test-=================
--------------------access zero address-----------------------
!!!read from reserved address 0x0...
External exception happened. MEXPFLAG[3]=0x4000000.
  Event 122: DMC_CMPA CPU memory protection fault for L1D (and other memory read
finally goes through the L1D controller)
  memory protection exception caused by local access at 0x0
  Supervisor Read violation
NRP=0xc0136a6, NTSR=0x1000d, TSCH= 0x1, TSCL= 0x2f6a969e
 B3=0xc013698, A4=0x25, B4= 0x28000c2, B14= 0x839fc0, B15= 0x838fa0

!!!write to reserved address 0x0...
External exception happened. MEXPFLAG[3]=0x10000000.
  Event 124: UMC_CMPA CPU memory protection fault for L2
  memory protection exception caused by local access at 0x0
  Supervisor Write violation
NRP=0xc0136e6, NTSR=0x1000d, TSCH= 0x1, TSCL= 0x2f6b83ef
 B3=0xc0136d4, A4=0x25, B4= 0x0, B14= 0x839fc0, B15= 0x838fa8
--------------------access address exceed LL2-----------------
!!!read from reserved address 0xa00000...
External exception happened. MEXPFLAG[3]=0x4000000.
  Event 122: DMC_CMPA CPU memory protection fault for L1D (and other memory read
finally goes through the L1D controller)
  memory protection exception caused by local access at 0xa00000
  Supervisor Read violation
NRP=0xc0136a6, NTSR=0x1000d, TSCH= 0x1, TSCL= 0x2f6c72f3
 B3=0xc013698, A4=0x2a, B4= 0x28000c2, B14= 0x839fc0, B15= 0x838fa0

!!!write to reserved address 0xa00000...
External exception happened. MEXPFLAG[3]=0x10000000.
  Event 124: UMC_CMPA CPU memory protection fault for L2
  memory protection exception caused by local access at 0xa00000
  Supervisor Write violation
NRP=0xc0136e6, NTSR=0x1000d, TSCH= 0x1, TSCL= 0x2f6d62b8
 B3=0xc0136d4, A4=0x2a, B4= 0xa00000, B14= 0x839fc0, B15= 0x838fa8
------------access local reserved configuration space----------
!!!read from reserved address 0x1000000...
External exception happened. MEXPFLAG[3]=0x4000000.
  Event 122: DMC_CMPA CPU memory protection fault for L1D (and other memory read
finally goes through the L1D controller)
  memory protection exception caused by local access at 0x1000000
  Supervisor Read violation
NRP=0xc0136a6, NTSR=0x1000d, TSCH= 0x1, TSCL= 0x2f6e550f
 B3=0xc013698, A4=0x2b, B4= 0x28000c2, B14= 0x839fc0, B15= 0x838fa0

!!!write to reserved address 0x1000000...
```

```
External exception happened. MEXPFLAG[3]=0x40000000.
  Event 126: EMC_CMPA CPU memory protection fault for local configuration space between
0x01000000-0x01BFFFFF
  memory protection exception caused by local access at 0x1000000
  Supervisor Write violation
NRP=0xc0136e4, NTSR=0x1000d, TSCH= 0x1, TSCL= 0x2f6f4561
 B3=0xc0136d4, A4=0x2b, B4= 0x1000000, B14= 0x839fc0, B15= 0x838fa8
---access reserved cfg space between INTC and power control----
!!!read from reserved address 0x18001c4...
External exception happened. MEXPFLAG[3]=0x4000000.
  Event 122: DMC_CMPA CPU memory protection fault for L1D (and other memory read
finally goes through the L1D controller)
  memory protection exception caused by local access at 0x18001c4
  Supervisor Read violation
NRP=0xc0136a6, NTSR=0x1000d, TSCH= 0x1, TSCL= 0x2f703bc7
 B3=0xc013698, A4=0x2b, B4= 0x18001c4, B14= 0x839fc0, B15= 0x838fa0

!!!write to reserved address 0x18001c4...
External exception happened. MEXPFLAG[3]=0x800000.
  Event 119: SYS_CMPA CPU Memory Protection Fault for local configuration of INTC and
power control
NRP=0xc0136e6, NTSR=0x1000d, TSCH= 0x1, TSCL= 0x2f712c2f
 B3=0xc0136d4, A4=0x2b, B4= 0x18001c4, B14= 0x839fc0, B15= 0x838fa8
------------access global reserved configration space----------
!!!read from reserved address 0x1cf0000...
External exception happened. MEXPFLAG[3]=0x84000000.
  Event 122: DMC_CMPA CPU memory protection fault for L1D (and other memory read
finally goes through the L1D controller)
  memory protection exception caused by local access at 0x1cf0000
  Supervisor Read violation
  Event 127: EMC_BUSERR Bus Error Interrupt for global configuration space between
0x01C00000 - 0x07FFFFFF
  CFG read status error detected
  XID (Transaction ID)= 12
  Addressing error
NRP=0xc0136a4, NTSR=0x1000d, TSCH= 0x1, TSCL= 0x2f71edc7
 B3=0xc013698, A4=0x2b, B4= 0x28000c2, B14= 0x839fc0, B15= 0x838fa0

!!!write to reserved address 0x1cf0000...
External exception happened. MEXPFLAG[3]=0x80000000.
  Event 127: EMC_BUSERR Bus Error Interrupt for global configuration space between
0x01C00000 - 0x07FFFFFF
  CFG write status error detected
  XID (Transaction ID)= 15
  Addressing error
NRP=0xc0136f0, NTSR=0x1000d, TSCH= 0x1, TSCL= 0x2f731225
 B3=0xc0136d4, A4=0x2b, B4= 0x1cf0000, B14= 0x839fc0, B15= 0x838fa8
-------------access reserved XMC configuration space-----------
!!!read from reserved address 0x8000100...
External exception happened. MEXPFLAG[3]=0x4004000.
  Event 110: MDMAERREVT XMC VBUSM error event
  MDMA read status error detected
  XID (Transaction ID)= 12
  Privilege error
  Event 122: DMC_CMPA CPU memory protection fault for L1D (and other memory read
finally goes through the L1D controller)
  memory protection exception caused by local access at 0x8000100
  Supervisor Read violation
NRP=0xc0136a4, NTSR=0x1000d, TSCH= 0x1, TSCL= 0x2f73fe47
 B3=0xc013698, A4=0x2b, B4= 0x28000c2, B14= 0x839fc0, B15= 0x838fa0
```

```
!!!write to reserved address 0x8000100...
External exception happened. MEXPFLAG[3]=0x4000.
  Event 110: MDMAERREVT XMC VBUSM error event
  memory protection exception caused by local access at 0x8000100
  Supervisor Write violation
NRP=0xc0136ea, NTSR=0x1000d, TSCH= 0x1, TSCL= 0x2f751dd9
 B3=0xc0136d4, A4=0x2b, B4= 0x8000100, B14= 0x839fc0, B15= 0x838fa8
-----------------access XMC unmapped space---------------------
!!!read from reserved address 0x1c000000...
External exception happened. MEXPFLAG[3]=0x4004000.
  Event 110: MDMAERREVT XMC VBUSM error event
  MDMA read status error detected
  XID (Transaction ID)= 6
  Privilege error
  Event 122: DMC_CMPA CPU memory protection fault for L1D (and other memory read
finally goes through the L1D controller)
  memory protection exception caused by local access at 0x1c000000
  Supervisor Read violation
NRP=0xc0136a4, NTSR=0x1000d, TSCH= 0x1, TSCL= 0x2f760f0c
 B3=0xc013698, A4=0x2c, B4= 0x28000c2, B14= 0x839fc0, B15= 0x838fa0

!!!write to reserved address 0x1c000000...
External exception happened. MEXPFLAG[3]=0x4000.
  Event 110: MDMAERREVT XMC VBUSM error event
  memory protection exception caused by local access at 0x1c000000
  Supervisor Write violation
NRP=0xc0136ea, NTSR=0x1000d, TSCH= 0x1, TSCL= 0x2f772e9b
 B3=0xc0136d4, A4=0x2c, B4= 0x1c000000, B14= 0x839fc0, B15= 0x838fa8
-----------------execute from reserved space-------------------
!!!execute at reserved address 0x0...
internal excpetion happened. IERR=0x1.
  Instruction fetch exception
NRP=0x0, NTSR=0x1000d, TSCH= 0x1, TSCL= 0x2f781ef7
 B3=0xc013744, A4=0x26, B4= 0x0, B14= 0x839fc0, B15= 0x838fa0
Exception happened at a place can not safely return!
```

7 参考文献

1. Memory Protection On KeyStone Devices (SPRWIKI9012)
2. TMS320C66x DSP CPU and Instruction Set Reference Guide (SPRUGH7)
3. TMS320C66x DSP CorePac User Guide (SPRUGW0)
4. KeyStone Architecture Multicore Shared Memory Controller User Guide (SPRUGW7)
5. KeyStone Architecture Memory Protection Unit (MPU) User Guide (SPRUGW5)
6. KeyStone Architecture Enhanced Direct Memory Access (EDMA3) Controller User Guide (SPRUGS5)
7. KeyStone Architecture Timer64 User Guide (SPRUGV5)
8. KeyStone Architecture Chip Interrupt Controller (CIC) User Guide (SPRUGW4)
9. KeyStone Architecture DDR3 Memory Controller's user guider (SPRUGV8)
10. TMS320C6000 Optimizing Compiler User's Guide (SPRU187)
11. "Interrupts", "MPU" and "Memory Map Summary" sections in Device specific Data Manuals

重要声明

德州仪器(TI) 及其下属子公司有权根据 JESD46 最新标准, 对所提供的产品和服务进行更正、修改、增强、改进或其它更改， 并有权根据 JESD48 最新标准中止提供任何产品和服务。客户在下订单前应获取最新的相关信息, 并验证这些信息是否完整且是最新的。所有产品的销售都遵循在订单确认时所提供的TI 销售条款与条件。

TI 保证其所销售的组件的性能符合产品销售时 TI 半导体产品销售条件与条款的适用规范。仅在 TI 保证的范围内，且 TI 认为 有必要时才会使用测试或其它质量控制技术。除非适用法律做出了硬性规定，否则没有必要对每种组件的所有参数进行测试。

TI 对应用帮助或客户产品设计不承担任何义务。客户应对其使用 TI 组件的产品和应用自行负责。为尽量减小与客户产品和应 用相关的风险，客户应提供充分的设计与操作安全措施。

TI 不对任何 TI 专利权、版权、屏蔽作品权或其它与使用了 TI 组件或服务的组合设备、机器或流程相关的 TI 知识产权中授予 的直接或隐含权限作出任何保证或解释。TI 所发布的与第三方产品或服务有关的信息，不能构成从 TI 获得使用这些产品或服 务的许可、授权、或认可。使用此类信息可能需要获得第三方的专利权或其它知识产权方面的许可，或是 TI 的专利权或其它 知识产权方面的许可。

对于 TI 的产品手册或数据表中 TI 信息的重要部分，仅在没有对内容进行任何篡改且带有相关授权、条件、限制和声明的情况 下才允许进行复制。TI 对此类篡改过的文件不承担任何责任或义务。复制第三方的信息可能需要服从额外的限制条件。

在转售 TI 组件或服务时，如果对该组件或服务参数的陈述与 TI 标明的参数相比存在差异或虚假成分，则会失去相关 TI 组件 或服务的所有明示或暗示授权，且这是不正当的、欺诈性商业行为。TI 对任何此类虚假陈述均不承担任何责任或义务。

客户认可并同意，尽管任何应用相关信息或支持仍可能由 TI 提供，但他们将独力负责满足与其产品及在其应用中使用 TI 产品 相关的所有法律、法规和安全相关要求。客户声明并同意，他们具备制定与实施安全措施所需的全部专业技术和知识，可预见 故障的危险后果、监测故障及其后果、降低有可能造成人身伤害的故障的发生机率并采取适当的补救措施。客户将全额赔偿因 在此类安全关键应用中使用任何 TI 组件而对 TI 及其代理造成的任何损失。

在某些场合中，为了推进安全相关应用有可能对 TI 组件进行特别的促销。TI 的目标是利用此类组件帮助客户设计和创立其特 有的可满足适用的功能安全性标准和要求的终端产品解决方案。尽管如此，此类组件仍然服从这些条款。

TI 组件未获得用于 FDA Class III（或类似的生命攸关医疗设备）的授权许可，除非各方授权官员已经达成了专门管控此类使 用的特别协议。

只有那些 TI 特别注明属于军用等级或“增强型塑料”的 TI 组件才是设计或专门用于军事/航空应用或环境的。购买者认可并同 意，对并非指定面向军事或航空航天用途的 TI 组件进行军事或航空航天方面的应用，其风险由客户单独承担，并且由客户独 力负责满足与此类使用相关的所有法律和法规要求。

TI 已明确指定符合 ISO/TS16949 要求的产品，这些产品主要用于汽车。在任何情况下，因使用非指定产品而无法达到 ISO/TS16949 要求，TI不承担任何责任。

	产品		应用
数字音频	www.ti.com.cn/audio	通信与电信	www.ti.com.cn/telecom
放大器和线性器件	www.ti.com.cn/amplifiers	计算机及周边	www.ti.com.cn/computer
数据转换器	www.ti.com.cn/dataconverters	消费电子	www.ti.com/consumer-apps
DLP® 产品	www.dlp.com	能源	www.ti.com/energy
DSP - 数字信号处理器	www.ti.com.cn/dsp	工业应用	www.ti.com.cn/industrial
时钟和计时器	www.ti.com.cn/clockandtimers	医疗电子	www.ti.com.cn/medical
接口	www.ti.com.cn/interface	安防应用	www.ti.com.cn/security
逻辑	www.ti.com.cn/logic	汽车电子	www.ti.com.cn/automotive
电源管理	www.ti.com.cn/power	视频和影像	www.ti.com.cn/video
微控制器 (MCU)	www.ti.com.cn/microcontrollers		
RFID 系统	www.ti.com.cn/rfidsys		
OMAP应用处理器	www.ti.com/omap		
无线连通性	www.ti.com.cn/wirelessconnectivity	德州仪器在线技术支持社区	www.deyisupport.com

邮寄地址： 上海市浦东新区世纪大道1568 号，中建大厦32 楼邮政编码： 200122