



SUNWAY 申威

申威 411 处理器核心 软件接口手册

2017 年 10 月

成都申威科技有限责任公司



免责声明

本档仅提供阶段性信息，所含内容可根据产品的实际情况随时更新，恕不另行通知。如因文档使用不当造成的直接或间接损失，本公司不承担任何责任。

成都申威科技有限责任公司

Chengdu Sunway Technology Corporation Limited

地址：成都市华府大道四段电子科大科技园 D22 栋

Building D22, National University Science and technology park,
Section 4, Huafu Avenue, Chengdu

Mail: sales@swcpu.cn

Tel : 028-68769016

Fax: 028-68769019

阅读指南

《申威 411 处理器核心软件手册》主要描述了申威 411 处理器的结构、处理器特权模式及程序、控制与状态寄存器描述、存储管理和结构、中断和异常描述、复位和初始化流程、事件计数、功耗管理、调试支持、I/O 虚拟化支持等内容。

文档修订

文档更新记录	文档名	申威 411 处理器核心软件接口手册
	版本号	V1.0
	创建人	研发部
	创建日期	2017-10-8

版本更新

版本号	更新内容	更新日期
V1.0	初稿	2017-10-8

技术支持

可通过邮箱或问题反馈网站向我司提交产品使用的问题，并获取技术支持。

售后服务邮箱：sales@swcpu.cn

问题反馈网址：<http://www.swcpu.cn/>

目 录

1	结构概述.....	1
1.1	核心结构.....	1
1.2	芯片结构.....	2
2	处理器模式与特权程序.....	3
2.1	处理器模式.....	3
2.2	特权程序.....	3
2.2.1	SYS_CALL 入口	4
2.2.2	异常和中断入口	5
2.3	特权指令.....	7
2.4	系统寄存器.....	7
3	控制与状态寄存器.....	8
3.1	CSR 的编址	8
3.2	CSR 访问机制	11
3.2.1	CSR 记分板.....	11
3.2.2	CSR 写机制.....	12
3.2.3	CSR 的存取顺序.....	12
3.3	指令部件中的 CSR	13
3.3.1	ITB_TAG.....	13
3.3.2	ITB_PTE.....	13
3.3.3	ITB_IA	13
3.3.4	ITB_IV	14
3.3.5	ITB_IVP	14
3.3.6	ITB_IU	14
3.3.7	ITB_IS	14
3.3.8	EXC_PC	14
3.3.9	IVA_FORM	15
3.3.10	PTBR.....	15
3.3.11	IER0~3.....	15
3.3.12	II_ER	16
3.3.13	II0	16
3.3.14	II1	16
3.3.15	II_MAIL	17
3.3.16	INT_STAT0~3	17
3.3.17	INT_VEC	18
3.3.18	INT_CLR0~3.....	18
3.3.19	II_CLR.....	18
3.3.20	EXC_SUM	19

3.3.21	PRI_BASE.....	20
3.3.22	IS_CTL.....	21
3.3.23	ISSUE_CTL.....	21
3.3.24	IS_STAT.....	22
3.3.25	IC_FLUSH.....	22
3.3.26	VPT_BASE.....	23
3.3.27	UPCR.....	23
3.3.28	PC0_CR.....	23
3.3.29	PC1_CR.....	24
3.3.30	VPCR.....	24
3.3.31	IA_MATCH.....	24
3.3.32	IA_MASK.....	25
3.3.33	HSERR_CNT.....	25
3.3.34	HSERR_TH.....	25
3.3.35	IGL_INST.....	26
3.3.36	SOFT_ERR.....	26
3.3.37	HARD_ERR.....	26
3.3.38	TIMER_CTL.....	27
3.3.39	TIMER_TH.....	27
3.4	数据 CACHE 管理部件中的 CSR.....	27
3.4.1	DTB_TAG.....	27
3.4.2	DTB_PTE.....	28
3.4.3	DTB_IA.....	28
3.4.4	DTB_IV.....	29
3.4.5	DTB_IVP.....	29
3.4.6	DTB_IU.....	29
3.4.7	DTB_IS.....	29
3.4.8	DTB_PCR.....	29
3.4.9	DS_STAT.....	30
3.4.10	DS_CTL.....	31
3.4.11	DC_CTL.....	31
3.4.12	DC_STAT.....	32
3.4.13	DA_MATCH.....	33
3.4.14	DA_MASK.....	33
3.4.15	DA_MATCH_MODE.....	33
3.4.16	DVA.....	34
3.4.17	DVA_FORM.....	34
3.5	二级 CACHE 管理部件中的 CSR.....	35
3.5.1	SC_CTL.....	35
3.5.2	SC_STAT.....	36
3.5.3	INT_REQ.....	38
3.5.4	CP_CTL.....	38
3.5.5	MERGE_OVTIME.....	39
3.6	整数部件中的 CSR.....	40

3.6.1	IVA.....	40
3.6.2	TC.....	40
3.6.3	TC_CTL.....	40
3.6.4	ICR.....	40
3.6.5	CID.....	41
3.6.6	IDA_MATCH.....	41
3.6.7	IDA_MASK.....	41
3.7	浮点运算部件中的 FPCR.....	42
3.7.1	FPCR 描述.....	42
3.7.2	FPCR 访问机制.....	45
4	存储管理和存储结构.....	46
4.1	存储空间.....	46
4.1.1	存储器空间.....	46
4.1.1.1	可 Cache 特性.....	46
4.1.1.2	不可 Cache 特性.....	47
4.1.1.3	访问顺序.....	49
4.1.2	I/O 空间.....	50
4.1.2.1	不可 Cache 特性.....	50
4.1.2.2	合并特性.....	50
4.1.2.3	访问顺序.....	50
4.2	CACHE 结构.....	51
4.2.1	指令 CACHE.....	51
4.2.2	数据 CACHE.....	52
4.2.3	二级 CACHE.....	52
4.2.4	指令副本标记.....	53
4.3	地址转换缓冲.....	53
4.3.1	指令流地址转换缓冲 (ITB).....	53
4.3.2	数据流地址转换缓冲 (DTB).....	54
4.4	存储空间访问方式.....	55
4.4.1	虚空间.....	55
4.4.2	访问方式.....	56
4.4.2.1	物理地址访问.....	56
4.4.2.2	虚地址访问.....	57
4.4.3	地址检查.....	57
4.4.4	指令 CACHE 访问.....	59
4.4.5	ITB 访问.....	59
4.4.6	DTB 访问.....	60
4.4.7	存储器栏栅 (MEMB).....	61
4.4.8	指令栏栅 (IMEMB).....	61
4.5	错误检测及错误处理.....	62
4.5.1	检错与报错.....	62
4.5.1.1	校验方法.....	62
4.5.1.2	可纠正错预警机制.....	63

4.5.1.3	核心的报错类型	64
4.5.2	指令 CACHE 的错误检测与处理	64
4.5.3	数据 CACHE 的错误检测与处理	65
4.5.4	二级 CACHE 的错误检测与处理	66
4.5.5	指令副本标记的错误检测与处理	66
4.5.6	核心接口上的错误检测与处理	66
4.5.7	核心响应中携带的错误	67
4.5.7.1	非法地址响应	67
4.5.7.2	带控制错的响应	68
4.5.7.3	带 ECC 多错的响应	68
4.5.8	核心内其它硬件故障的检测与处理	69
4.5.8.1	指令流控制错	69
4.5.8.2	数据流控制错	69
5	中断	70
5.1	中断源	70
5.1.1	核间中断	71
5.1.1.1	核间中断 0/1	71
5.1.1.2	核间异步消息中断	71
5.1.2	核内产生的中断	72
5.1.2.1	已纠正错中断	72
5.1.2.2	机器检查错中断	73
5.1.2.3	定时器中断	74
5.1.2.4	事件计数中断	74
5.1.3	核外产生的中断	74
5.1.3.1	PCI-E 的 INTx 中断	75
5.1.3.2	PCI-E 的 MSI 中断	76
5.1.3.3	短时钟中断	76
5.1.3.4	维护中断	77
5.1.3.5	故障中断	77
5.1.4	特殊中断	77
5.1.4.1	睡眠中断	77
5.1.4.2	睡眠唤醒中断	78
5.1.4.3	复位	79
5.2	中断的优先级	79
5.3	中断的嵌套	79
6	异常	81
6.1	DTB 脱靶	81
6.2	ITB 脱靶	81
6.3	浮点屏蔽故障	82
6.4	不对界故障	82
6.5	数据流故障	82

6.6	指令流故障.....	83
6.7	操作码非法.....	83
6.8	算术自陷.....	84
6.8.1	整数溢出.....	84
6.8.2	无效操作.....	85
6.8.3	下溢.....	85
6.8.4	上溢.....	85
6.8.5	除数为零.....	86
6.8.6	非精确结果.....	86
6.8.7	原子操作溢出.....	86
7	复位和初始化.....	88
7.1	复位的种类.....	88
7.2	上电复位.....	89
7.3	冷复位.....	89
7.4	睡眠复位.....	90
7.4.1	睡眠流程.....	90
7.4.2	唤醒流程.....	90
7.5	复位后的 CSR.....	91
8	事件计数.....	93
8.1	事件计数相关 CSR.....	93
8.2	事件计数的模式.....	93
8.3	事件计数的流程.....	96
9	功耗管理.....	98
9.1	核心深睡眠.....	98
9.2	核心浅睡眠.....	98
9.3	关闭浮点部件时钟.....	98
9.4	动态功耗调整.....	98
10	调试支持.....	100
10.1	指令流调试支持.....	100
10.1.1	SYS_CALL 指令.....	100
10.1.2	指令流地址匹配.....	100
10.1.3	指令流跳转目标地址匹配.....	100
10.2	数据流调试支持.....	101
11	I/O 虚拟化支持.....	102
11.1	段式地址代换.....	102
11.2	页式地址代换.....	102

11.2.1 设备表	103
11.2.2 IO 页表	104
附录 A CSR 的限制	105
附录 B 事件计数内容与方法	107
附录 C CHIP_ID 定义	111
附录 D 各种中断的初始向量值	112
附录 E 对软件的特殊要求	113

1 结构概述

申威 411 处理器为 64 位字长的 RISC 架构多核处理器，采用多核结构和片上系统技术，在单芯片中集成了 4 个同构的新一代申威 Core3 核心（简称 C3 核心）、6MB 的三级共享 Cache、两路 64 位 DDR3 存储控制器、两路标准 PCI-E 2.0 接口以及自定义的维护接口。

1.1 核心结构

申威 411 处理器采用新一代申威 Core3 核心，基本结构如图 1-1 所示：

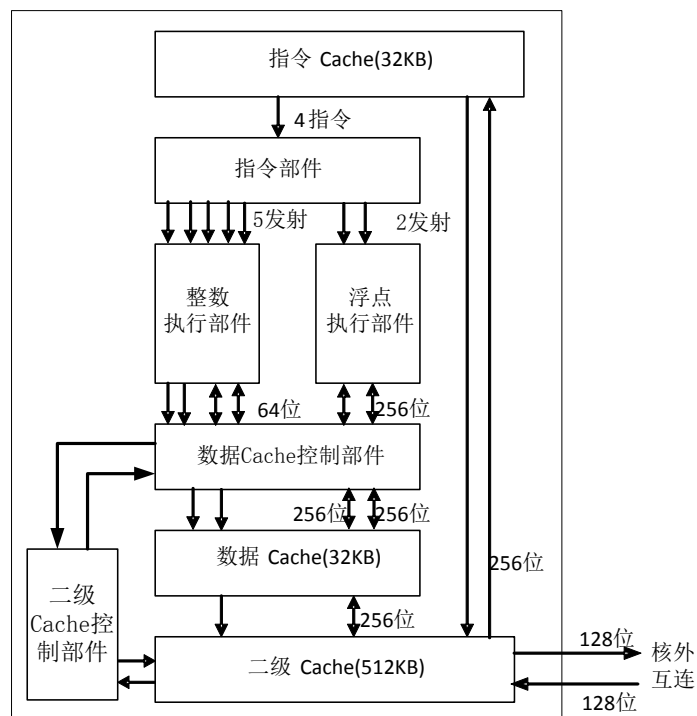


图 1-1: 申威 411 处理器核心结构图

Core3 由指令部件、整数执行部件、浮点执行部件、数据 Cache 控制部件、二级 Cache 控制部件以及一级指令 Cache、一级数据 Cache 和二级 Cache 组成。其技术特征如下：

- 1) 核心为采用并行发射、乱序发射、乱序执行和推测执行技术的 4 译码 7 发射超标量结构；
- 2) 采用短向量加速计算技术提高整数和浮点运算性能，支持浮点双 256 位 SIMD 流水线、整数单 256 位 SIMD 流水线，每个时钟周期可产生 11 个整数运算结果或 16 个浮点运算结果；
- 3) 一级指令 Cache 容量为 32KB，采用四路组相联结构，虚地址访问方式，Cache 行大小为 128 字节，采用可容错的偶校验；
- 4) 一级数据 Cache 容量为 32KB，采用四路组相联结构，物理地址访问方式，Cache 行大小为 128 字节，采用可纠错的 ECC 校验；
- 5) 二级 Cache 容量为 512KB，采用八路组相联结构，物理地址访问方式，Cache 行大小为 128 字节，为指令和数据混合 Cache，采用可纠错的 ECC 校验；

- 6) 一级数据 Cache 与二级 Cache 为严格的包含关系，一级指令 Cache 与二级 Cache 为既不包含，也不互斥关系，硬件自动支持指令与数据的 Cache 一致性。

1.2 芯片结构

申威 411 处理器芯片包含四个 Core3 核心、6MB 的三级共享 Cache 和两路 DDR3 存储器接口、两路 PCI-E 接口以及其它维护测试接口，其结构图如下所示：

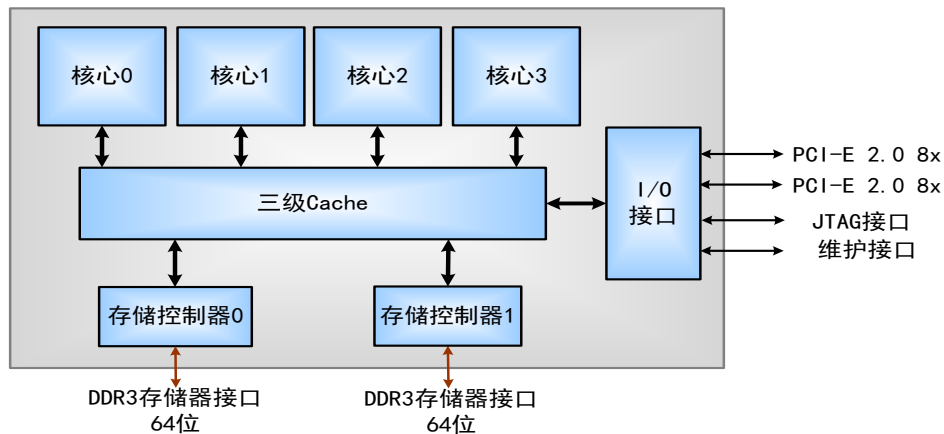


图 1-2: 申威 411 处理器芯片结构图

其特点有：

- 1) 单芯片集成 4 个核心，采用 40nm 代工工艺，工作频率最高可达 1.6GHz；
- 2) 采用新一代 C3 结构核心技术，核心流水线升级为 4 译码 7 发射结构，单核性能大幅提升（整数性能提高 62%，浮点性能提高 53%）；
- 3) 功耗较低，1.6GHz 下的典型运行功耗为 25~30W；
- 4) 集成 6MB 的三级共享 Cache，可大幅提升访存性能；
- 5) I/O 性能指标优异，并支持更加丰富的 I/O 虚拟化；
- 6) 集成的两路存储控制器和两路 PCI-E 接口，都可根据应用需求对接口数量进行裁剪，以降低系统开发成本，降低功耗；
- 7) 提供两种封装产品：LGA 封装和 BGA 封装；LGA 封装版本与申威 410 引脚兼容，方便用户升级设计。

2 处理器模式与特权程序

2.1 处理器模式

申威 411 处理器设置了 4 种处理器模式 (Processor Mode: PM)，不同的核心可以同时处于不同的模式，具体定义如下：

1) 硬件模式 (Hardware Mode: HM)：最高级模式，为特权程序运行环境。此模式下，可使用特权指令直接对底层硬件进行控制和操作，可使用系统寄存器。指令流只能以物理地址访问，不进行虚实地址转换，数据流可以用物理地址访问，也可以用虚地址访问，由具体的访存指令决定。所有的中断被屏蔽，不响应任何中断请求；

2) 虚拟模式 (Virtual Mode: VM)：次高级模式，介于硬件与操作系统之间，提供虚拟层服务。此模式下，不能使用特权指令和系统寄存器，指令流和数据流可采用包括超页模式在内的虚地址访问，由访问地址[42]位决定，允许响应中断请求；

3) 内核模式 (Kernel Mode: KM)：次低级模式，为操作系统运行环境。此模式下，不能使用特权指令和系统寄存器，指令流和数据流都采用虚地址访问（不支持超页模式），允许响应中断请求；

4) 用户模式 (User Mode: UM)：最低级模式，为用户程序运行环境。此模式下，不能使用特权指令和系统寄存器，指令流和数据流都采用虚地址访问（不支持超页模式），允许响应中断请求。

程序计数器 PC 的最低两位用于记录当前处理器模式值，为“00₂”表示硬件模式 (HM)，为“01₂”表示虚拟模式 (VM)，为“10₂”表示内核模式 (KM)，为“11₂”表示用户模式 (UM)。申威 411 处理器中不同的核心可以运行在不同的模式下，处理器模式有以下几种转换方式：

- 1) 非硬件模式下执行 SYS_CALL 指令，可以从非硬件模式转换到硬件模式；
- 2) 核心响应中断请求时，可以从非硬件模式转换到硬件模式；
- 3) 核心发现异常时，可以从非硬件模式转换到硬件模式；
- 4) 硬件模式下执行 PRI_RET 指令，可以从硬件模式转换到任意模式。

2.2 特权程序

特权程序是针对特定操作系统的一组程序，只能在 HM 模式下运行。在特权程序中，可使用五条特权指令，读写处理器控制与状态寄存器，进行存储器管理等操作。

特权程序可以由以下事件触发：

- 1) 中断，包括复位、内部中断、外部中断和机器检查错等；
- 2) 异常，包括算术自陷和存储管理故障等；
- 3) SYS_CALL 指令。

特权程序主要应用于以下几种情况：

- 1) 一些必要的功能用硬件实现过于复杂，但又无法用常规程序来实现。如 TLB 装填、中断确认和异常处理等；
- 2) 一些必须为原子操作的功能，实现的指令序列较长，且需要访问底层硬件。如从异常或中断返回等；
- 3) 一些为实现不同处理器之间的兼容或者减少编程工作量的功能，不经常使用，由专门硬件来实现的代价过大。

特权程序在指定的入口被调用，申威 411 处理器有两种特权程序入口，一种是 SYS_CALL 指令相关的特权程序入口，另一种是异常和中断相关的特权程序入口。

2.2.1 SYS_CALL 入口

在 VM、KM 和 UM 模式下，执行 SYS_CALL 指令时，会将指令流转到特权程序入口执行，入口地址由基址寄存器 (PRI_BASE) 和指令功能码决定。SYS_CALL 指令的返回地址由系统寄存器 SR23 来保存，其中 SR23[1:0] 记录运行 SYS_CALL 指令时的处理器模式。

SYS_CALL 指令含一个功能码，需要对功能码的合法性进行检查，若其功能码满足以下任何一个条件，则产生非法操作码异常：

- 1) UM 模式下，SYS_CALL 指令只能启动用户级特权程序，如果对应的指令功能码[7:0]位不在 0x80~0xBF (用户级特权程序) 范围内，则产生非法操作码异常；
- 2) KM 模式下，SYS_CALL 指令可以启动用户级和内核级特权程序，如果对应的指令功能码[7:0]位不在 0x00~0x3F (内核级特权程序) 和 0x80~0xBF (用户级特权程序) 范围内，则产生非法操作码异常；
- 3) VM 模式下，SYS_CALL 指令可以启动用户级、内核级和虚拟级特权程序，如果对应的指令功能码[7:0]位不在 0x00~0x3F (内核级特权程序)、0x80~0xBF (用户级特权程序) 和 0x40~0x7F (虚拟级特权程序) 范围内，则产生非法操作码异常；
- 4) UM、KM、VM 和 HM 模式下，SYS_CALL 指令功能码[25:8]位不为全“0”，则产生非法操作码异常。

在 HM 模式下，SYS_CALL 指令功能码[25:8]位为全“0”则当作空指令。SYS_CALL 指令的功能码与其对应的特权程序入口地址 (PC) 的计算方法如下：

- 1) PC[63:40]=全“0”；
- 2) PC[39:15]={PRI_BASE[39:16], 0₂}；
- 3) PC[14:13]与功能码[7:6]的对应关系如表 2-1；

表 2-1: SYS_CALL 指令部分功能码与特权程序入口关系表

功能码[7:6]	PC[14:13]
----------	-----------

10 ₂	11 ₂
00 ₂	10 ₂
01 ₂	01 ₂

- 4) PC[12:7]=功能码[5:0];
- 5) PC[6:2]=全“0”;
- 6) PC[1:0]=“00₂”(进入 HM 模式的标志)。

由此可见，虚拟级特权程序起始偏移地址为 0x2000，内核级特权程序起始偏移地址为 0x4000，用户级特权程序起始偏移地址为 0x6000。2 条功能码相邻的 SYS_CALL 指令的特权程序入口地址间距为 128 字节，即可以包含连续的 32 条指令，对于指令数量超过 32 条的特权程序，必须跳转到另一地址继续处理。

2.2.2 异常和中断入口

当硬件检测到没有屏蔽的异常或中断时，会自动跳到对应的特权程序入口进行处理。触发异常或被中断的指令 PC 值保存在异常地址寄存器 (EXC_PC) 中。表 2-2 列出了各种异常入口相对于基址寄存器 (PRI_BASE) 的偏移，表中各种异常或中断按优先级从低到高依次排列。

表 2-2: 异常或中断服务特权程序入口偏移地址

序号	入口名称	类型	偏移	描述
1	DTBM_DOUBLE	故障	0x800	DTB 二次脱靶。指示 PRI_LDx/vpte 指令进行数据流虚实地址转换时产生的 DTB 脱靶。
2	DTBM_SINGLE_0	故障	0x880	DTB 用户脱靶。指示非 PRI_LDx/vpte 指令进行数据流虚实地址转换时产生的 DTB 脱靶且虚地址 VA[42]为“0”。
3	DTBM_SINGLE_1	故障	0x900	DTB 核心脱靶。指示非 PRI_LDx/vpte 指令进行数据流虚实地址转换时产生的 DTB 脱靶且虚地址 VA[42]为“1”。
4	ITB_MISS_0	故障	0x980	ITB 用户脱靶。指示指令流进行虚实地址转换时产生的 ITB 脱靶且虚地址 VA[42]为“0”。
5	ITB_MISS_1	故障	0xA00	ITB 核心脱靶。指示指令流进行虚实地址转换时产生的 ITB 脱靶且虚地址 VA[42]为“1”。
6	FEN	故障	0xA80	浮点屏蔽故障。指示浮点屏蔽时运行浮点或 SIMD 扩展指令产生的故障。
7	UNALIGN	故障	0xB00	不对界异常。指示数据流访问地址不对界产生的故

				障。
8	ARITH	算术自陷	0xB80	算术自陷。指示浮点或整数运算指令产生异常且对应的算术自陷使能。
9	保留	—	0xC00	
10	DFAULT	故障	0xC80	数据流故障。指示数据流地址越界、访问越权、数据流地址匹配、虚地址符号检查错、数据 Cache 的标记偶校验错、数据 Cache 数据校验错、存储器数据多错等故障。
11	IACV	故障	0xD00	指令流故障。指示指令流访问越权、虚地址符号检查错、指令流地址匹配或目标地址匹配。
12	OPCDEC	故障	0xD80	操作码非法故障。指示出现非法的操作码或功能码。
13	INTERRUPT0	中断	0xE00	一般中断 0。指示产生了核间中断。
14	INTERRUPT1	中断	0xE80	一般中断 1。指示产生了核间中断之外的普通硬件中断。
15	SLEEP	中断	0xF00	睡眠中断。指示产生了请求核心进入睡眠状态的中断。
16	MCHK	中断	0xF80	机器检查错中断。指示核心内部产生不能恢复的故障，精确和不精确的机器检查错，都进入该入口。
17	RESET/WAKEUP	中断	0x1000	复位/睡眠唤醒中断。指示核心进入复位或从睡眠状态被唤醒。

上表中，RESET/WAKEUP 具有最高优先级，总是无条件中止正常指令流，进入对应的特权程序入口执行。MCHK、SLEEP、INTERRUPT1、INTERRUPT0 中断的优先级依次降低，且都只能在指令边界上被识别，产生这些中断时，必须等前面所有指令都正常退出后，才能进入对应的特权程序入口执行。

除中断以外，其它入口都与具体的指令执行相关。在指令退出时，检查指令执行过程中是否产生异常，如果产生，则进入对应的特权程序入口执行。如果同一条指令在流水线上的不同站台产生多个异常，则优先处理先产生的异常，忽略后产生的异常。如果同一条指令在流水线上的同一站台产生多个异常，则按上表中的优先级，优先处理优先级高的异常，忽略优先级低的异常。

如浮点指令或 SIMD 扩展指令可同时产生 IACV、FEN 和 OPCDEC 异常，这时就要按照优先级，优先处理 OPCDEC 异常，无 OPCDEC 异常时再处理 IACV 异常，无 OPCDEC 异常和 IACV 异常时，再处理 FEN 异常。转移指令可能同时发生转移地址符号扩展错（BAD_IVA）和转移目标地址匹配（IDA_MATCH），此时都进指令流故障故障（IACV），但优先报地址符号扩展错，只登记 CSR：EXC_SUM[BAD_IVA]。

基于基址寄存器（PRI_BASE）的 0x0~0x7FF 范围内存储器空间，没有作为任何特权程序的入

口,此空间可以作为特权程序的数据空间或者信息保留空间。0x1000~0x1FFF 范围内的存储器空间,都可作为 RESET/WAKEUP 特权程序存放空间。

2.3 特权指令

特权指令与申威 411 处理器的具体实现有关,用于访问核心的控制与状态寄存器 CSR,进行存储管理等。特权指令只能在 HM 模式下执行,在其它模式下执行将产生非法操作码异常。特权指令共有 5 条,如表 2-3 所示,具体的指令定义参看《申威 411 处理器指令系统手册》。

表 2-3: 特权指令一览

序号	指令	操作码	描述
1	PRI_LD	0x25	特权装入指令
2	PRI_ST	0x2D	特权存储指令
3	PRI_RET	0x07	特权程序返回指令。
4	PRI_RCSR	0x06.FE _{xx}	将控制与状态寄存器 CSR 内容读到整数寄存器
5	PRI_WCSR	0x06.FF _{xx}	将整数寄存器内容写到控制与状态寄存器 CSR

2.4 系统寄存器

申威 411 处理器包含 10 个特殊的寄存器,用作特权程序的临时数据存储空间,被称作系统寄存器 (SR)。当申威 411 处理器处于 HM 模式且指令流控制寄存器 (IS_CTL) 的 SRE 位为“1”时,整数结构寄存器中的 R1、R2、R4~R7 和 R20~R23 被识别为系统寄存器。运行 SYS_CALL 指令时默认使用系统寄存器 R23 来保存 SYS_CALL 指令的返回地址。其它情况下,则不允许使用系统寄存器。

3 控制与状态寄存器

核心设置了一组处理器控制与状态寄存器（CSR），用于实现对核心的控制，记录核心的运行状态。CSR 只能在 HM 模式下，通过特权指令 PRI_RCSR 和 PRI_WCSR 进行读写操作。

为简化对 CSR 描述，使用一些符号来对 CSR 内容的软件访问特性和复位状态进行描述，其中

- 1) “W” 指示可写，“WO” 指示只写；
- 2) “R” 指示可读，“RO” 指示只读；
- 3) “RW” 指示可读写；
- 4) “W1C” 指示写“1”清除，写“0”无影响；
- 5) “WC” 指示写清除；
- 6) “RO,x” 指示只读且复位时初值为“x”；
- 7) “WO,x” 指示只写且复位时初值为“x”；
- 8) “RW,x” 指示可读写且复位时初值为“x”；
- 9) “SEXT(x)” 指示将“x”符号扩展指定位数。

以上“x”可以为“0”，也可以为“1”，没有指定复位设置值的 CSR 在复位后是一个不确定的值，需要在复位后进行初始化时设置。读保留位时，返回“0”；写保留位时，写入的值被忽略。

3.1 CSR 的编址

通过特权指令 PRI_RCSR 和 PRI_WCSR 来读写 CSR 时，这两条指令的[7:0]位作为 CSR 访问的地址，下表 3-1、3-2、3-3、3-4 分别列出所有 CSR 的名称、符号、缺省使用的记分板位和存取特性等。

表 3-1: 指令部件中 CSR

序号	名称	符号	索引	记分板位	存取特性
1	ITB 的 TAG 写寄存器	ITB_TAG	0000 0000	0	WO
2	ITB 的 PTE 写寄存器	ITB_PTE	0000 0001	0	WO
3	ITB 全部刷新寄存器	ITB_IA	0000 0010	0	WO
4	ITB 虚拟机刷新寄存器	ITB_IV	0000 0011	0	WO
5	ITB 虚拟机部分刷新寄存器	ITB_IVP	0000 0100	0	WO
6	ITB 用户进程刷新寄存器	ITB_IU	0000 0101	0	WO
7	ITB 条目刷新寄存器	ITB_IS	0000 0110	0	WO
8	指令虚地址格式寄存器	IVA_FORM	0000 0111	0	RO

9	PTBR 页表基地址寄存器	PTBR	0000 1000	0	RW
10	核间中断使能寄存器	II_ER	0000 1001	0	RW
11	核间中断 0 寄存器	II0	0000 1010	0	RO
12	异常摘要寄存器	EXC_SUM	0000 1101	0	RO
13	异常地址寄存器	EXC_PC	0000 1110	0	RO
14	清除核间中断寄存器	II_CLR	0000 1111	0	WO
15	特权程序基地址寄存器	PRI_BASE	0001 0000	0	RW
16	指令流控制寄存器	IS_CTL	0001 0001	0	RW
17	发射控制寄存器	ISSUE_CTL	0001 0011	0	RW
18	指令流状态寄存器	IS_STAT	0001 0100	0	R, WIC
19	指令 Cache 刷新寄存器	IC_FLUSH	0001 0101	0	WO
20	虚页表基地址寄存器	VPT_BASE	0001 0111	0	RW
21	虚拟机控制寄存器	VPCR	0001 1001	0	RW
22	指令流地址匹配寄存器	IA_MATCH	0001 1010	0	RW
23	指令流地址屏蔽寄存器	IA_MASK	0001 1011	0	RW
24	核间中断 1 寄存器	II1	0001 1100	0	RO
25	核间异步消息中断寄存器	II_MAIL	0001 1101	0	RO
26	中断向量映射寄存器	INT_VEC	0001 1111	0	RW
27	用户进程控制寄存器	UPCR	0010 00nn	0	W
			0010 00xx	0	R
28	事件计数与控制寄存器 0	PC0_CR	0010 0100	0	RW
29	事件计数与控制寄存器 1	PC1_CR	0010 0101	0	RW
30	核心可纠正错计数器	HSERR_CNT	0010 0110	0	RW
31	计数阈值寄存器	HSERR_TH	0010 0111	0	RW
32	非法指令寄存器	IGL_INST	0010 1000	0	RO
33	软件故障寄存器	SOFT_ERR	0010 1001	0	RW
34	定时器控制寄存器	TIMER_CTL	0010 1010	0	RW
35	定时器阈值寄存器	TIMER_TH	0010 1011	0	RW
36	中断状态寄存器 0	INT_STAT0	0011 0000	0	RO
37	中断状态寄存器 1	INT_STAT1	0011 0001	0	RO
38	中断状态寄存器 2	INT_STAT2	0011 0010	0	RO
39	中断状态寄存器 3	INT_STAT3	0011 0011	0	RO
40	清除普通硬件中断寄存器 0	INT_CLR0	0011 0100	0	WO

41	清除普通硬件中断寄存器 1	INT_CLR1	0011 0101	0	WO
42	清除普通硬件中断寄存器 2	INT_CLR2	0011 0110	0	WO
43	清除普通硬件中断寄存器 3	INT_CLR3	0011 0111	0	WO
44	普通硬件中断使能寄存器 0	IER0	0011 1000	0	RW
45	普通硬件中断使能寄存器 1	IER1	0011 1001	0	RW
46	普通硬件中断使能寄存器 2	IER2	0011 1010	0	RW
47	普通硬件中断使能寄存器 3	IER3	0011 1011	0	RW
48	硬件故障寄存器	HARD_ERR	0011 1100	0	RW

表 3-2: 数据 Cache 管理部件中 CSR

序号	名称	符号	索引	记分板位	存取特性
1	DTB 的 TAG 写寄存器	DTB_TAG	0100 0000	1	WO
2	DTB 的 PTE 写寄存器	DTB_PTE	0100 0001	1	WO
3	DTB 全部刷新寄存器	DTB_IA	0100 0010	1	WO
4	DTB 虚拟机刷新寄存器	DTB_IV	0100 0011	1	WO
5	DTB 虚拟机部分刷新寄存器	DTB_IVP	0100 0100	1	WO
6	DTB 用户进程刷新寄存器	DTB_IU	0100 0101	1	WO
7	DTB 刷新寄存器	DTB_IS	0100 0110	1	WO
8	DTB 进程号寄存器	DTB_PCR	0100 0111	1	RW
9	数据流状态寄存器	DS_STAT	0100 1000	1	R,WC
10	数据流控制寄存器	DS_CTL	0100 1001	1	RW
11	数据 Cache 控制寄存器	DC_CTL	0100 1010	1	RW
12	数据 Cache 状态寄存器	DC_STAT	0100 1011	1	R,W1C
13	数据流地址匹配寄存器	DA_MATCH	0100 1100	1	RW
14	数据流地址屏蔽寄存器	DA_MASK	0100 1101	1	RW
15	数据虚地址寄存器	DVA	0100 1110	1	RO
16	数据虚地址格式寄存器	DVA_FORM	0100 1111	1	RO
17	数据流地址匹配模式寄存器	DA_MATCH_MODE	0101 0000	1	RW

表 3-3: 二级 Cache 管理部件中 CSR

序号	名称	符号	索引	记分板位	存取特性
1	二级 Cache 控制寄存器	SC_CTL	1000 0000	2	RW
2	二级 Cache 状态寄存器	SC_STAT	1000 0001	2	R,W1C
3	中断请求寄存器	INT_REQ	1000 0010	2	RW
4	Cache 一致性控制寄存器	CP_CTL	1000 0011	2	RO

5	合并缓冲超时控制寄存器	MERGE_OVTIME	1000 0100	2	RW
---	-------------	--------------	-----------	---	----

表 3-4: 整数部件中 CSR

序号	名称	符号	索引	记分板位	存取特性
1	指令虚地址寄存器	IVA	1100 0000	3	RO
2	计时器	TC	1100 0001	3	RW
3	周期计数控制寄存器	TC_CTL	1100 0010	3	RW
4	整数控制寄存器	ICR	1100 0011	3	RW
5	核心识别码寄存器	CID	1100 0100	3	RW
6	指令流目标地址寄存器	IDA_MATCH	1100 0101	3	RW
7	指令流目标地址屏蔽寄存器	IDA_MASK	1100 0110	3	RW

注意：对没有定义的CSR进行读写，不产生任何故障与异常，即PRI_WCSR写指令不产生任何影响，PRI_RCSR读指令读出内容是不可预测的。浮点部件中的FPCR属于特殊的CSR，可以在任何模式下通过非特权指令RFPCR、WFPCR、SETFPEC0、SETFPEC1、SETFPEC2、SETFPEC3进行读写访问。

3.2 CSR 访问机制

CSR 既可被软件读写，也可被核心硬件读写，对 CSR 的访问可分为以下四类：

- 1) 显式读 CSR：通过 PRI_RCSR 指令读出 CSR 中的内容；
- 2) 隐式读 CSR：硬件根据 CSR 中的内容执行相应的操作，如进行指令流地址虚实代换时，就是对 ITB 的隐式读；
- 3) 显式写 CSR：通过 PRI_WCSR 指令修改 CSR 中的内容；
- 4) 隐式写 CSR：硬件执行过程中根据当前的状态更新 CSR 中的内容，如 Load 指令产生数据流故障而修改 EXC_PC、DVA、DS_STAT 等 CSR；接到中断请求时，更新 INT_STAT 等 CSR。

为保证隐式和显示读写 CSR 的正确性，申威 411 处理器在具体实现时设置了相应的访问控制机制，但在某些特殊情况下需要软件来保证。

3.2.1 CSR 记分板

在申威411处理器实现中，硬件设置了4个CSR记分板位，来控制访问不同模块内的CSR，维持CSR访问顺序的正确性，其中[0]位对应指令部件内的CSR，[1]位对应数据Cache管理内的CSR，[2]位对应二级Cache管理部件内的CSR，[3]位对应整数部件内的CSR。CSR记分板的应用规则如下：

- 1) PRI_WCSR指令发射时要求所访问的CSR使用的记分板位已被清除；
- 2) PRI_WCSR指令在发射后将所访问的CSR使用的记分板位置“1”，在该指令完成处理并退

出或被中止时，将所访问的CSR使用的记分板位清除；

- 3) PRI_RCSR指令发射时要求所访问的CSR使用的记分板位已被清除；
- 4) 访存指令发射时要求CSR记分板[1]位和[2]位都被清除。

3.2.2 CSR 写机制

CSR 中的内容直接控制处理器的运行，因此申威 411 处理器必须保证 CSR 不会被推测执行的指令修改。在执行显示写 CSR 操作时，硬件只能在 PRI_WCSR 指令退出时，才能修改对应的 CSR 内容。在发生异常等产生隐式写 CSR 的请求时，硬件只能在异常指令退出时，将该指令相关的异常信息写入对应的 CSR 中。

3.2.3 CSR 的存取顺序

下面分别说明 CSR 四种访问方式下各种组合之间的顺序关系。描述方式采用“方式 1→方式 2”，其中方式 1 是先发生的访问 CSR 方式，方式 2 后发生的访问 CSR 方式。

- 1) 隐式读→隐式读：允许两个读操作被重排序；
- 2) 隐式读→隐式写：不存在这种情况；
- 3) 隐式读→显式读：允许两个读操作被重排序；
- 4) 隐式读→显式写：由 CSR 写机制来保证顺序性；
- 5) 隐式写→隐式读：不存在这种情况；
- 6) 隐式写→隐式写：由 CSR 写机制来保证顺序性；
- 7) 隐式写→显式读：这类都是与异常和中断相关的 CSR 访问，由 HM 模式下访问机制保证；
- 8) 隐式写→显式写：这类都是与异常和中断相关的 CSR 访问，由 HM 模式下访问机制保证；
- 9) 显式读→隐式读：允许两个读操作被重排序；
- 10) 显式读→隐式写：由 CSR 写机制来保证顺序性；
- 11) 显式读→显式读：允许两个读操作被重排序；
- 12) 显式读→显式写：由 CSR 记分板机制保证；
- 13) 显式写→隐式读：通常情况下 CSR 记分板机制可以保证，特殊情况下需要软件协助保证；
- 14) 显式写→隐式写：通常情况下 CSR 写机制可以保证，特殊情况下需要软件协助保证；
- 15) 显式写→显式读：由 CSR 记分板机制保证顺序性；
- 16) 显式写→显式写：由 CSR 记分板机制保证顺序性。

需要软件协助保证CSR访问顺序的要求，详见附录A。

3.3 指令部件中的 CSR

3.3.1 ITB_TAG

ITB_TAG 为 ITB 的虚页号写寄存器，只写，装填 ITB 时，先将虚页号写入该寄存器，随后写 CSR: ITB_PTE 寄存器时，才将该寄存器内容一并写入 ITB 中。

表3-5: ITB_TAG寄存器域描述

名称	范围	类型	描述
VA[42:13]	[42:13]	WO	装入ITB条目中的虚页号。
—	其它位	—	保留。

3.3.2 ITB_PTE

ITB_PTE 为 ITB 的 PTE 写寄存器，只写，装填 ITB 时，写该寄存器，将 CSR: ITB_TAG、VPCR[VPN]、UPCR[UPN]和该寄存器内容一并写入 ITB。

表3-6: ITB_PTE寄存器域描述

名称	范围	类型	描述
PFN[39:13]	[39:13]	WO	物理页面号，ITB装填时，实际只写入[38: 13]位。
UEE	[11]	WO	用户模式可执行位，为“1”表示指令流可在用户模式下访问该页面。
KEE	[10]	WO	核心模式可执行位，为“1”表示指令流可在核心模式下访问该页面。
VEE	[9]	WO	虚拟模式可执行位，为“1”表示指令流可在虚拟模式下访问该页面。
GH[1:0]	[6:5]	WO	页面粒度，值为0、1、2或3，分别指示该页为1、32、1024或32768个连续的8KB页组成的大页（分别对应8KB、256KB、8MB或256MB地址对界的连续空间）。
KS	[4]	WO	系统空间标志，为“0”表示该页面为用户空间，为“1”表示该页面为系统空间。
—	其它位	—	保留。

3.3.3 ITB_IA

ITB_IA 为 ITB 全部刷新寄存器，只写，写该寄存器不需要数据，将清除 ITB 中所有条目。

3.3.4 ITB_IV

ITB_IV 为 ITB 虚拟机刷新寄存器，只写，写该寄存器不需要数据，将清除 ITB 中 VPN 与 CSR: VPCR[VPN]相同的所有条目。

3.3.5 ITB_IVP

ITB_IVP 为 ITB 虚拟机部分刷新寄存器，只写，写该寄存器不需要数据，将清除 ITB 中 VPN 与 CSR: VPCR[VPN]相同且 KS 为“0”的所有条目。

3.3.6 ITB_IU

ITB_IU 为 ITB 用户进程刷新寄存器，只写，写该寄存器不需要数据，将清除 ITB 中 VPN 与 CSR: VPCR[VPN]相同，且 UPN 与 CSR: UPCR[UPN]相同，且 KS 为“0”的所有条目。

3.3.7 ITB_IS

ITB_IS 为 ITB 条目刷新寄存器，只写，写该寄存器可刷新 ITB 中满足以下条件之一的条目：

1) ITB 条目中虚页面号与所写 CSR: ITB_IS 数据[42:13]位匹配（要根据条目中页面粒度 GH 来确定实际比较的地址位数）、VPN 与 CSR: VPCR[VPN]相同、UPN 与 CSR: UPCR[UPN]相同且 KS 为“0”；

2) ITB 条目的虚页面号与所写 CSR: ITB_IS 数据[42:13]位匹配（要根据条目中页面粒度 GH 来确定实际比较的地址位数）、VPN 与 CSR: VPCR[VPN]相同，且 KS 为“1”。

表3-7: ITB_IS寄存器域描述

名称	范围	类型	描述
INVL_ITB[42:13]	[42:13]	WO	需要刷新的虚页号。
—	其它位	—	保留。

3.3.8 EXC_PC

EXC_PC 为异常地址寄存器，只读，用于保存异常或中断时的指令流虚地址，它由硬件隐式修改。

表3-8: EXC_PC寄存器域描述

名称	范围	类型	描述
----	----	----	----

SEXT (PC[42])	[63:43]	RO	指令流虚地址[42]位的符号扩展。
PC[42:2]	[42:2]	RO	指令流虚地址。
PM	[1:0]	RO	异常或中断产生时的处理器模式。

该寄存器记录的地址有两种含义：

- 1) 如果是异常，则该寄存器记录的是产生异常的指令 PC 值；
- 2) 如果是中断，则该寄存器记录的是中断处理完成后准备执行的指令 PC 值。

3.3.9 IVA_FORM

IVA_FORM 为指令虚地址格式寄存器，只读，该寄存器读出的内容来自 CSR: EXC_PC 和 VPT_BASE，目的是为了便于软件处理 ITB 脱靶异常。

表3-9: IVA_FORM寄存器域描述

名称	范围	类型	描述
SEXT (VPT_BASE[42])	[63:43]	RO	虚页表基地址[42]位的符号扩展。
VPT_BASE[42:39]	[42:39]	RO	虚页表基地址。
SEXT(EXC_PC[42])	[38:33]	RO	异常指令虚页号[42]位的符号扩展。
EXC_PC[42:13]	[32:3]	RO	异常指令虚页号。
—	[2:0]	—	保留。

3.3.10 PTBR

PTBR 为页表基地址寄存器，读写，用于存放一级页表的基址，方便软件处理二次脱靶。

表3-10: PTBR寄存器域描述

名称	范围	类型	描述
—	其它	—	保留。
PTBR_PA[39:0]	[39:0]	RW, 0	一级页表基址。

3.3.11 IER0~3

IER0~3为普通硬件中断请求使能寄存器，可读写，该寄存器用于控制CSR: INT_STAT0~3中所有中断的使能（最多支持64种中断请求的屏蔽），与INT_STAT0~3一一对应。当该寄存器中某一位为“1”且对应的中断请求有效则进入中断请求处理。复位以后，各种中断在INT_STAT0~3中对应的位置见附录D。

3.3.12 II_ER

II_ER为核间中断使能寄存器。

表3-11: II_ER寄存器域描述

名称	范围	类型	描述
—	其它	—	保留。
II_MAIL_EN	[2]	RW	处理器内部各核心之间发送的异步消息中断使能，当为“1”时，如果收到异步消息的数量大于0，则允许进入中断处理。
II1_EN	[1]	RW	处理器内部各核心之间发送的核间中断1使能，当为“1”时，如果收到核间中断1，则允许进入中断处理。
II0_EN	[0]	RW	处理器内部各核心之间发送的核间中断0使能，当为“1”时，如果收到核间中断0，则允许进入中断处理。

3.3.13 II0

II0为核间中断0寄存器，只读，用于记录核心（含本核心）发向本核心的核间中断0请求有效位。在上电复位或冷复位时清除，睡眠复位时内容保持不变。本寄存器记录的核间中断请求与对应的核间中断使能（II_ER[II0_EN]）无关，显式读此寄存器可获得本核心收到的所有的核间中断0请求，有效位为“1”，则说明收到了核间中断0请求；有效位为“0”，则说明没有收到核间中断0请求。

表3-12: II0寄存器域描述

名称	范围	类型	描述
—	其它位	—	保留
II0Valid[3:0]	[3:0]	RO,0	核心（含本核心）向本核心发出的核间中断0请求有效位，当第i位为“1”时，表示第i个核心向本核心发的核间中断0请求有效。

3.3.14 II1

II1为核间中断1寄存器，只读，用于记录核心（含本核心）发向本核的核间中断1请求有效位。在上电复位或冷复位时清除，睡眠复位时内容保持不变。本寄存器记录的核间中断请求与对应的核间中断使能（II_ER[II1_EN]）无关，显式读此寄存器可获得本核心收到的所有的核间中断1请求，有效位为“1”，则说明收到了核间中断1请求；有效位为“0”，则说明没有收到核间中断1请求。

表3-13: II1寄存器域描述

名称	范围	类型	描述
—	其它位	—	保留
II1Valid[3:0]	[3:0]	RO,0	核心（含本核心）向本核心发出的核间中断类型1请求有效位，当第i位为“1”时，表示第i个核心向本核心发的核间中断1请求有效。

3.3.15 II_MAIL

II_MAIL 为核间异步消息中断寄存器，只读，用于记录核心（含本核心）发向本核心的异步消息数量。在上电复位或冷复位时清除，睡眠复位时内容保持不变。

表3-14: II_MAIL寄存器域描述

名称	范围	类型	描述
—	其它	—	保留。
MAIL_NUM	[5:0]	RO, 0	是本核心收到的异步消息数量的补码表示，最高位为符号位。当MAIL_NUM大于“0”时，如果II_ER[II_MAIL_EN]为“1”，则产生异步消息中断请求。

注：II_MAIL 是核心对收到的异步消息数量的计数，其它核心向本核心发送一个异步消息时，该计数器加“1”，本核心成功读走一个异步消息时，该计数器减“1”，由于允许软件随意读异步消息，因此该计数器的值可能出现负数，但负数只是一种中间状态，当对异步消息的读写操作停止一段时间后，该计数器的值一定为正数或“0”。

3.3.16 INT_STAT0~3

INT_STAT0~3 为中断状态寄存器，只读，用于记录申威 411 处理器产生的、对应中断使能有效的所有普通硬件中断请求，共 256 个。INT_STAT0 对应 0~63 号中断，INT_STAT1 对应 64~127 号中断，INT_STAT2 对应 128~191 号中断，INT_STAT3 对应 192~255 号中断。该寄存器记录睡眠中断、机器检查错中断、已纠正中断、事件计数中断 1、事件计数中断 0 和所有外部中断，每位具体代表哪种中断由核心内部的 CSR：INT_VEC 或核心外部的部分 I/O 寄存器（IOR）决定（IOR 的地址和类型可参看《申威 411 处理器 IO 寄存器手册》）。在上电复位或冷复位时清“0”，睡眠复位

时内容保持不变，复位后各种中断在 INT_STAT0~3 中对应的位置见附录 D。

本组寄存器记录的普通硬件中断请求与对应的普通硬件中断使能（IER0~3）相关，显式读此寄存器只能获得使能打开的普通硬件中断请求信息，对应位为“1”，则说明收到了对应类型的中断请求，并且该类型的中断使能打开；对应位为“0”，则可能是收到了对应类型的中断请求，并且该类型的中断使能关闭，也可能是没有收到对应类型的中断请求。

3.3.17 INT_VEC

INT_VEC 为中断向量映射寄存器，可读写，用于设置事件计数中断、已纠正中断、机器检查错中断、睡眠中断在 INT_STAT0~3 中位置。在上电复位或冷复位时进行设置，睡眠复位时内容保持不变。

表3-15: INT_VEC寄存器域描述

名称	范围	类型	描述
—	其它	—	保留。
TIMER_VEC[7:0]	[47:40]	RW,0xFA	指定定时器中断对应的向量。
PC0_VEC[7:0]	[39:32]	RW,0xFB	指定事件计数0中断对应的向量。
PC1_VEC[7:0]	[31:24]	RW,0xFC	指定事件计数1中断对应的向量。
CR_VEC[7:0]	[23:16]	RW,0xFD	指定已纠正中断对应的向量。
MCHK_VEC[7:0]	[15:8]	RW,0xFE	指定机器检查错中断对应的向量。
SLEEP_VEC[7:0]	[7:0]	RW,0xFF	指定睡眠中断对应的向量。

3.3.18 INT_CLR0~3

INT_CLR0~3 为清除普通硬件中断寄存器，只写，用于清除 INT_STAT0~3 中对应的普通硬件中断请求，与 INT_STAT0~3 一一对应。写该寄存器时，若对应的数据位为“1”，则 INT_STAT0~3 中对应的中断请求被清除。复位以后，各种中断在 INT_STAT0~3 中对应的位置见附录 D。

3.3.19 II_CLR

II_CLR 为清除核间中断寄存器，只写，用于清除 II0 和 II1 中记录的核间中断请求有效位。

表3-16: II_CLR寄存器域描述

名称	范围	类型	描述
—	其它	—	保留。
II1_CLR[15:0]	[31:16]	WO	清除核间中断1寄存器，写该寄存器的某位为“1”

			时，值表示将CSR: II1[II1Valid]中的某位清“0”。
II0_CLR[15:0]	[15:0]	WO	清除核间中断0寄存器，写该寄存器的某位为“1”时，值表示将CSR: II0[II0Valid]中的某位清“0”。

3.3.20 EXC_SUM

EXC_SUM 为异常摘要寄存器，只读，用于存放产生异常指令的相关信息，硬件在产生异常的指令退出时更新此 CSR，仅当在异常处理特权程序的第一个指令组读该寄存器，得到的内容才是准确、有效的。该寄存器保存下列三种异常相关的信息：

1) 算术自陷：指令产生了必须报告给操作系统的异常情况，该寄存器的 REG 域包含触发自陷的指令目标寄存器号；

2) 指令流故障：产生了指令流目标地址与指定地址匹配成功、指令流地址与指定地址匹配成功，以及跳转目标指令地址符号扩展错，但不包括 ITB 访问引起的指令流地址越权。当 IDA_MATCH 或 BAD_IVA 有效时，该寄存器的 REG 域包含跳转指令的 Ra 域；

3) 数据流故障：数据流访问产生了 DTB 脱靶、地址不对界以及其它数据流故障，该寄存器 REG 域包含触发故障的访存指令 Ra 域。

表3-17: EXC_SUM寄存器域描述

名称	范围	类型	描述	
IDA_MATCH	[34]	RO	指示整数转移指令的目标地址与指定地址匹配成功。	
IA_MATCH	[33]	RO	指示指令流地址与指定地址匹配成功。	
“0”	[32]	RO	固定为“0”。	浮点向量第四个元素运算产生的浮点算术自陷。
INE3	[31]	RO	指示浮点扩展部件产生浮点不精确结果自陷。	
UNF3	[30]	RO	指示浮点扩展部件产生浮点下溢自陷。	
FOV3	[29]	RO	指示浮点扩展部件产生浮点上溢自陷。	
DZE3	[28]	RO	指示除数为“0”自陷。	
INV3	[27]	RO	指示浮点扩展部件产生无效操作自陷。	
“0”	[26]	RO	固定为“0”	
INE2	[25]	RO	指示浮点扩展部件产生浮点不精确结果自陷。	
UNF2	[24]	RO	指示浮点扩展部件产生浮点下溢自陷。	
FOV2	[23]	RO	指示浮点扩展部件产生浮点上溢自陷。	
DZE2	[22]	RO	指示除数为“0”自陷。	
INV2	[21]	RO	指示浮点扩展部件产生无效操作自陷。	浮点向量第二个元素运算产
“0”	[20]	RO	固定为“0”	
INE1	[19]	RO	指示浮点扩展部件产生浮点不精确结果自陷。	

UNF1	[18]	RO	指示浮点扩展部件产生浮点下溢自陷。	生的算术自陷。
FOV1	[17]	RO	指示浮点扩展部件产生浮点上溢自陷。	
DZE1	[16]	RO	指示除数为“0”自陷。	
INV1	[15]	RO	指示浮点扩展部件产生无效操作自陷。	
INCO	[14]	RO	指示处理“取并加1”或“取并减1”指令时，产生整数溢出自陷。	
BAD_IVA	[13]	RO	指示跳转类指令的跳转目标指令地址产生符号扩展错。	
REG[4:0]	[12:8]	RO	产生异常的Load/Store指令或跳转指令的Ra域，或者运算类指令的目标寄存器域（Ra或Rc或Rd）。若产生异常的是ITB脱靶、中断、OPCDEC、IA_MATCH、非Load、Store指令和运算类指令，则该域的值是不可预测的。	
INT	[7]	RO	指示整数部件产生整数溢出自陷。	
OVI0	[6]	RO	指示浮点转换为整数或整数运算时产生溢出自陷。	浮点或浮点向量第一个元素运算产生的算术自陷。
INE0	[5]	RO	指示不精确结果自陷。	
UNF0	[4]	RO	指示下溢自陷。	
FOV0	[3]	RO	指示上溢自陷。	
DZE0	[2]	RO	指示除数为“0”自陷。	
INV0	[1]	RO	指示无效操作自陷。	
SWC	[0]	RO	指示需要操作系统异常处理程序对出现异常的浮点指令进行仿真处理（即在FPCR[1]位为“0”时产生浮点异常）。	
—	其它位	—	保留。	

3.3.21 PRI_BASE

PRI_BASE 为特权程序基地址寄存器，可读写，用于保存特权程序的物理基地址，在上电复位或冷复位时按照核心所在核组进行设置，睡眠复位时内容保持不变。

表3-18: PRI_BASE寄存器域描述

名称	范围	类型	描述
PRI_BASE[39:16]	[39:16]	RW	特权程序的物理基地址，[39]固定为“0”，上电复位和冷复位时[36:35]位设置为核组的编号，其它位为“0”。申威411相当于只有一个核组，故该域为0。
—	其它位	—	保留。

3.3.22 IS_CTL

IS_CTL 为指令流控制寄存器，可读写，该寄存器用于控制指令部件的各种功能，复位时设置为“0x10088”。

表3-19: IS_CTL寄存器域描述

名称	范围	类型	描述
HM_PEN	[16]	RW,1	硬件模式下进行预测使能信号，为“1”表示可以进行预测。
PC1_EN	[15]	RW,0	事件计数器1使能。
PC0_EN	[14]	RW,0	事件计数器0使能。
PC_CM	[13:12]	RW,0	指定在何种模式下进行事件计数：“00 ₂ ”为硬件模式；“01 ₂ ”为虚拟模式；“10 ₂ ”为内核模式；“11 ₂ ”为用户模式。
CM_PCE	[11]	RW,0	指定模式事件计数使能。
AM_PCE	[10]	RW,0	任何模式事件计数使能。
BIST_FAIL	[9]	RO,0	BISR自修复或BIST自测试结果：为“0”表示BISR或BIST成功，否则表示失败。
CHIP_ID[5:0]	[8:3]	RO	申威411处理器芯片的ID号，值为0x38，具体定义详见附录C。
F_BAD_DPAR	[2]	RW,0	指令Cache装填时，强制装填错误的的数据校验位。
F_BAD_TPAR	[1]	RW,0	指令Cache装填时，强制装填错误的标记地址校验位。
SRE	[0]	RW,0	系统寄存器使能：若SRE为“1”，则在HM模式下可访问系统寄存器。
—	其它位	—	保留。

其中，PC0_EN、PC1_EN、AM_PCE、CM_PCE、PC_CM 是事件计数相关的控制位，当 PC0_EN 为“1”，且 AM_PCE 为“1”或 CM_PCE 为“1”时，事件计数器 0 中断使能；当 PC1_EN 为“1”，且 AM_PCE 为“1”或 CM_PCE 为“1”时，事件计数器 1 中断使能。AM_PCE 和 CM_PCE 不能同时为“1”，如果同时为“1”，则按所有模式进行事件计数。当 CM_PCE 为“1”时，硬件只对 PC_CM 中指定的模式下发生的事件计数。事件计数的具体方法详见 8.3 和附录 B。

3.3.23 ISSUE_CTL

ISSUE_CTL 为指令发射控制寄存器，可读写，ISSUE_INT[9:0]控制整数指令发射，发射站台判断最高位若为“1”，则表示此周期允许整数指令发射，否则禁止整数指令发射，ISSUE_INT[9:0]中的值在每个周期都进行循环左移。ISSUE_FLT[9:0]作用与 ISSUE_INT[9:0]相同，不同在于控制浮点指令发射。此寄存器在上电复位或冷复位时设置为 0xFFF，睡眠复位时内容保持不变。

表3-20: ISSUE_CTL寄存器域描述

名称	范围	类型	描述
ISSUE_FLT[9]	[19]	RO,1	固定为“1”。
ISSUE_FLT[8:0]	[18:10]	RW,0x1FF	浮点发射控制位。
ISSUE_INT[9]	[9]	RO,1	固定为“1”。
ISSUE_INT[8:0]	[8:0]	RW,0x1FF	整数发射控制位。
—	其它位	—	保留。

3.3.24 IS_STAT

IS_STAT 为指令流状态寄存器，可读写，用于保存指令流发现的各种错误状态，通过写“1”可清除对应的状态。

表3-21: IS_STAT寄存器域描述

名称	范围	类型	描述
BTB_PAR	[6]	R,W1C	转移预测阵列偶校验错，指令流访问转移预测阵列时发生偶校验错。（此位在申威411处理器中没有实现，保留位）
MEM_MERR	[5]	R,W1C	指令流访问主存时，主存数据产生ECC多错。
DATA_PAR	[4]	R,W1C	指令Cache的数据偶校验错，指示在取指时读指令Cache数据存储器产生偶校验错。
TAG_PAR	[3]	R,W1C	指令Cache的标记偶校验错，指示在取指时读指令Cache标记存储器产生偶校验错。
ROB_ERR	[2]	R,W1C	重排序缓冲状态非法，指示重排序缓冲（ROB）的头尾指针管理混乱。
ADDR_ERR	[1]	R,W1C	指令流数据响应错，指示从主存取指令时，因指令流访问不存在的地址空间而返回数据错响应。
ACK_ERR	[0]	R,W1C	指令流引起访存响应错，指示从主存取指令时，访存部件收到的响应和请求类型不匹配。
—	其它位	—	保留。

3.3.25 IC_FLUSH

IC_FLUSH 为指令 Cache 刷新寄存器，只写，写不需要数据。写该寄存器时将清除整个指令 Cache，刷新过程中，不再进行新的取指令操作，直到刷新完成。

3.3.26 VPT_BASE

VPT_BASE 寄存器为虚页表基地址寄存器，可读写。

表3-22: VPT_BASE寄存器域描述

名称	范围	类型	描述
SEXT(VPTB[42])	[63:43]	RO	虚页表基地址[42]位的符号扩展。
VPTB[42:39]	[42:39]	RW	虚页表基地址。
—	其它位	—	保留。

3.3.27 UPCR

UPCR为用户进程控制寄存器，可读写，记录进程的相关信息。读总是返回寄存器所有位域的信息，写既可以修改整个寄存器，也可以只修改此寄存器中某个域，CSR索引号[0]位为“1”指示写SCE、FPE和FCE域，CSR索引号[1]位为“1”指示写UPN域。复位时设置为“0x07”。

表3-23: UPCR寄存器域描述

名称	范围	类型	CSR索引	描述
其它	—	—	—	保留。
UPN[7:0]	[46:39]	RW,0	[1]位	用户进程号。
SCE	[2]	RW,1	[0]位	SIMD从流水线时钟使能，若为“0”，则关闭SIMD从流水线时钟。
FPE	[1]	RW,1	[0]位	浮点使能，若为“0”，则不允许执行浮点指令。
FCE	[0]	RW,1	[0]位	浮点部件时钟使能，若为“0”，则关闭浮点部件时钟。

3.3.28 PC0_CR

PC0_CR为事件计数与控制寄存器0，可读写。用于读写事件计数器0的计数值和设置事件计数器0的计数条件。

表3-24: PC0_CR寄存器域描述

名称	范围	类型	描述
PC0_SEL[3:0]	[63:60]	RW	事件计数器0的计数对象选择，具体定义详见8.2
—	[59:58]	—	保留。

PC0[57:0]	[57:0]	RW	事件计数器0。
-----------	--------	----	---------

3.3.29 PC1_CR

PC1_CR为事件计数与控制寄存器1，可读写。用于读写事件计数器1的计数值和设置事件计数器1的计数条件。

表3-25: PC1_CR寄存器域描述

名称	范围	类型	描述
PC1_SEL[5:0]	[63:58]	RW	事件计数器1的计数对象选择，具体定义详见8.2
PC1[57:0]	[57:0]	RW	事件计数器1。

3.3.30 VPCR

VPCR为虚拟机进程控制寄存器，可读写，记录虚拟机的相关信息。复位时设置为全“0”。

表3-26: VPCR寄存器域描述

名称	范围	类型	描述
VPN	[45:44]	RW	虚拟机号。
VM_ST	[43:0]	RW	虚拟模式状态位。
—	其它位	—	保留。

3.3.31 IA_MATCH

IA_MATCH 为指令流地址匹配寄存器，可读写，复位时设置为全“0”。该寄存器用于指令流调试，当指令流访问地址与该寄存器内容匹配时，将产生指令流故障。

表3-27: IA_MATCH寄存器域描述

名称	范围	类型	描述
—	其它	—	保留。
EN	[53]	RW,0	匹配使能信号，为“1”表示匹配使能。
VPN[1:0]	[52:51]	RW,0	指定需要匹配的虚拟机号。
UPN[7:0]	[50:43]	RW,0	指定需要匹配的用户进程号。
ADDR[42:0]	[42:0]	RW,0	指定需要匹配的指令虚地址，其中[1:0]位指定需要匹配的处理器模式。

3.3.32 IA_MASK

IA_MASK 为指令流地址屏蔽寄存器，可读写，复位时为“0x1F,FFFF,FFFF,FFFF”。该寄存器用于指令流调试，屏蔽进行指令流地址比较的位数。

表3-28: VPCR寄存器域描述

名称	范围	类型	描述
—	其它位	—	保留。
MASK[52:0]	[52:0]	RW,全“1”	指令流地址匹配屏蔽位，对应IA_MATCH[52:0]中的每一位，为“1”表示该位参与比较。

3.3.33 HSERR_CNT

HSERR_CNT 为核心可纠正错计数器，复位时清“0”。当核心内的 ICACHE、DCACHE 和 SCACHE、转移预测（针对该阵列的功能申威 411 处理器没有实现）存储阵列发生可纠正错时，该寄存器的已纠正错计数器 HSERR_CNT 加“1”；当时间计数器溢出时，将该计数器清“0”。当已纠正计数器溢出时，将 SERR_OF 置为“1”，如果已纠正错中断使能打开，则产生已纠正错中断。判断溢出的条件由核心已纠正错阈值寄存器 HSERR_TH 决定。软件写该寄存器时，如果第[0]位为“1”，则 HSERR_CNT[31:0]清“0”；如果第[32]位为“1”，则 SERR_OF 清“0”。

表3-29: HSERR寄存器域描述

名称	范围	类型	描述
—	其它位	—	保留。
SERR_OF	[32]	RO,W1C,0	核心已纠正错计数器溢出标志。
HSERR_CNT[31:0]	[31:0]	RO,W1C,0	核心已纠正错计数器的数值。

3.3.34 HSERR_TH

HSERR_TH 为已纠正错计数器阈值寄存器，复位时已纠正错计数器阈值置为“0x00”，时间计数器阈值置为“0x1F”。该寄存器的 SERR_THLD[4:0]用于控制已纠正错计数器的溢出条件，即指示这 32 位计数器中某位有进位时，则表示已该计数器溢出。该寄存器的 TIME_THLD[5:0]用于控制时间计数器的溢出条件，即分别指示这 64 位计数器中某位有进位时，则表示已该计数器溢出。TIME_THLD[6]为时间计数屏蔽位，为“1”时，时间计数器不计数。

表3-30: HSERR_TH寄存器域描述

名称	范围	类型	描述
—	其它位	—	保留。

SERR_THLD[4:0]	[11:7]	RW,0	核心已纠正错计数器的阈值，指示已纠正错计数器产生溢出的条件。
TIME_THLD[6:0]	[6:0]	RW,0x1F	时间计数器的阈值。最高位为时间计数屏蔽位，低6位指示时间计数器产生溢出的条件。

3.3.35 IGL_INST

IGL_INST 为非法指令寄存器，只读，复位时，清 0。当指令流出现非法操作码故障时，该寄存器表示非法指令的 32 位信息。

表3-31: IGL_INST寄存器域描述

名称	范围	类型	描述
—	其它位	—	保留。
LOCK_IGL	[32]	RO, 0x0	锁指令非法标志，当该位为“1”表示： 1) 译码站台收到LSTx指令时，如果LSTx指令不在指令0位置，或者指令1位置上不是RD_F指令，则LSTx指令报操作码非法故障； 2) 译码站台收到RD_F指令时，如果指令0位置上不是LSTx指令，或者RD_F指令不在指令1位置，则RD_F指令报操作码非法故障。
IGL_INST [31:0]	[31:0]	R0,0x0	非法指令信息。

3.3.36 SOFT_ERR

SOFT_ERR 为软件故障标志寄存器，软件可读写，复位时，清 0。该寄存器置从“0”变为“1”时，可向维护接口报告一个软件故障，该故障可产生一个故障中断，也可通过芯片引脚向系统报错。

表3-32: SOFT_ERR寄存器域描述

名称	范围	类型	描述
—	其它位	—	保留。
SOFT_ERR	[0]	RW,0x0	软件故障标志位，只能由软件来设置。

3.3.37 HARD_ERR

HARD_ERR 为硬件故障标志寄存器，软件可读写，复位时，清 0。该寄存器置从“0”变为“1”

时，可向维护接口报告一个硬件故障，该故障可产生一个故障中断，也可通过芯片引脚向系统报错。

表3-33: HARD_ERR寄存器域描述

名称	范围	类型	描述
—	其它位	—	保留。
HARD_ERR	[0]	RW,0x0	硬件故障标志位，只能由软件来设置。

3.3.38 TIMER_CTL

TIMER_CTL 为核心定时器控制寄存器，软件可读写，复位时清 0，产生定时器中断时自动清 0。

表 3-34: TIMER_CTL 寄存器域描述

名称	范围	类型	描述
—	其它位	—	保留
TIMER_EN	[0]	RW, 0	定时器使能信号，只有在该信号为1时，内部定时器才开始从0开始进行加1计数，直至到达TIMER_TH指定的阈值。

3.3.39 TIMER_TH

TIMER_TH 为核心定时器阈值寄存器，软件可读写，复位时清 0。

表 3-35: TIMER_TH 寄存器域描述

名称	范围	类型	描述
TIMER_TH	[63:0]	RW, 0	定时器阈值，当CSR: TIMER_CTL[TIMER_EN]为“1”时，计数器就从0开始加“1”计数，加到所设阈值以后便产生定时器中断。

3.4 数据 Cache 管理部件中的 CSR

3.4.1 DTB_TAG

DTB_TAG 为 DTB 的虚页号写寄存器，只写，装填 DTB 时，先将虚页号写入该寄存器，随后写 CSR: DTB_PTE 寄存器时，才将该寄存器内容一并写入 DTB 中。

表3-29: DTB_TAG寄存器域描述

名称	范围	类型	描述
----	----	----	----

VA[42:13]	[42:13]	WO	数据流访问地址虚页号。
—	其它位	—	保留。

3.4.2 DTB_PTE

DTB_PTE 为 DTB 的 PTE 写寄存器，只写，装填 DTB 时，写该寄存器，将 CSR: DTB_TAG、DTB_PCR[VPN、UPN]和该寄存器内容一并写入 DTB。

表3-30: DTB_PTE寄存器域描述

名称	范围	类型	描述
PA[39:13]	[58:32]	WO	数据流访问地址物理页面号。
UWE	[15]	WO	UM可写位，为“1”时，在UM模式下可对该页面执行写操作。
KWE	[14]	WO	KM可写位，为“1”时，在KM模式下可对该页面执行写操作。
VWE	[13]	WO	VM可写位，为“1”时，在VM模式下可对该页面执行写操作。
HWE	[12]	WO	HM可写位，为“1”时，在HM模式下可对该页面执行写操作。
URE	[11]	WO	UM可读位，为“1”时，在UM模式下可对该页面执行读操作。
KRE	[10]	WO	KM可读位，为“1”时，在KM模式下可对该页面执行读操作。
VRE	[9]	WO	VM可读位，为“1”时，在VM模式下可对该页面执行读操作。
HRE	[8]	WO	HM可读位，为“1”时，在HM模式下可对该页面执行读操作。
GH[1:0]	[6:5]	WO	页面粒度，值为0、1、2或3，分别对应1、32、1024或32768个连续且对界的8KB页（分别对应8KB、256KB、8MB或256MB地址对界的连续空间）。
KS	[4]	WO	系统空间标志，为“0”表示该页面为用户空间，为“1”表示该页面为系统空间。
FOW	[2]	WO	写故障位，为“1”指示对此页面的写访问将产生数据流故障（DFAULT）。
FOR	[1]	WO	读故障位，为“1”指示对此页面的读访问将产生数据流故障（DFAULT）。
—	其它位	—	保留。

3.4.3 DTB_IA

DTB_IA为DTB全部刷新寄存器，只写，写不需要数据。写该寄存器将所有DTB的条目置为无效。

3.4.4 DTB_IV

DTB_IV为DTB虚拟机刷新寄存器，只写，写该寄存器不需要数据，将清除DTB中所有VPN与CSR: DTB_PCR[VPN]相同的所有条目。

3.4.5 DTB_IVP

DTB_IVP为DTB虚拟机部分刷新寄存器，只写，写该寄存器不需要数据，将清除DTB中VPN与CSR: DTB_PCR[VPN]相同且KS为“0”的所有条目。

3.4.6 DTB_IU

DTB_IU为DTB用户进程刷新寄存器，只写，写该寄存器不需要数据，将清除DTB中VPN与CSR: DTB_PCR[VPN]相同，且UPN与CSR: DTB_PCR[UPN]相同，且KS为“0”的所有条目。

3.4.7 DTB_IS

DTB_IS为DTB条目刷新寄存器，只写。写该寄存器可刷新DTB中满足以下条件之一的条目：

1) DTB条目的虚页面号与该寄存器的VA[42:13]位匹配（要根据条目中的页面粒度GH来确定实际匹配的地址位数），且VPN域与CSR: DTB_PCR[VPN]相同，且UPN域与CSR: DTB_PCR[UPN]相同，且KS为0；

2) DTB条目的虚页面号与该寄存器的VA[42:13]位匹配（要根据条目中的页面粒度GH来确定实际匹配的地址位数），VPN域与CSR: DTB_PCR[VPN]相同，且KS为“1”。

表3-31: DTB_IS寄存器域描述

名称	范围	类型	描述
VA[42:13]	[42:13]	WO	需要刷新DTB条目的虚页号。
—	其它位	—	保留。

3.4.8 DTB_PCR

DTB_PCR为DTB进程控制寄存器，可读写。该寄存器用于设置DTB装填和刷新时的虚拟机号、用户进程号，以及HM模式下的虚地址数据流访问指令查询DTB时使用的替代处理器模式。

表3-32: DTB_PCR寄存器域描述

名称	范围	类型	描述
----	----	----	----

VPN[1:0]	[63:62]	RW	虚拟机号
UPN[7:0]	[61:54]	RW	用户进程号
PM[1:0]	[1:0]	RW	处理器模式。
—	其它位	—	保留。

3.4.9 DS_STAT

DS_STAT 为数据流状态寄存器，可读写，写不需要数据，直接将其内容清“0”。该寄存器用于记录数据流发生地址符号扩展错、SIMD 对界错、校验错或存储管理故障时的相关状态和指令信息。注意，当标量指令发生不对界错时，会清除所有标志，该错误的处理不依赖于这个寄存器的内容；当产生 DTBM_DOUBLE 异常自陷时，不修改此寄存器内容。

表3-33: DS_STAT寄存器域描述

名称	范围	类型	描述
—	其它位	—	保留。
SCD_SERR	[17]	R,WC	访存指令对二级Cache执行读操作时，数据产生ECC单错。由于该错误是通过硬件重发流程进行纠错的，因此软件应该忽略此标志。
MEM_MERR	[16]	R,WC	对存储器执行读操作时，主存数据产生ECC多错。
SIMD_MV	[15]	R,WC	指示一般SIMD访存指令的地址不对界（对指令访问的向量数据而言不对界，但对每个向量元素访问粒度而言是对界的）。
DAT_ERR	[14]	R,WC	指示访存指令读数据Cache的数据阵列时产生ECC校验错，作为数据流故障。
DA_MATCH	[13]	R,WC	指示数据流地址匹配成功，作为数据流故障。
ACV1	[12]	R,WC	指示数据流访问地址超出主存容量、读访问不存在的I/O寄存器、SIMD访存指令、原子操作指令或不可Cache访存指令访问I/O空间地址、锁指令访问IO空间或不可Cache空间，作为数据流故障。
BAD_DVA	[11]	R,WC	指示计算数据流虚地址时产生符号扩展错。
TAG_PERR	[10]	R,WC	指示访存指令读数据Cache的标记阵列时产生偶校验错，作为数据流故障。
OPCODE[5:0]	[9:4]	R,WC	指示引起DTB脱靶、不对界异常或数据流故障的指令操作码。
FOW	[3]	R,WC	指示写访问一个具有FOW属性的页面而产生写故障。
FOR	[2]	R,WC	指示读访问一个具有FOR属性的页面而产生读故障。
ACV0	[1]	R,WC	指示在DTB虚实地址转换过程中出现存取越权。

WR	[0]	R,WC	指示写访问产生DTB脱靶、不对界异常或数据流故障。
----	-----	------	---------------------------

3.4.10 DS_CTL

DS_CTL为数据流控制寄存器，可读写，存放数据流访问的控制信息。复位时为全“0”。

表3-34: DS_CTL寄存器域描述

名称	范围	类型	描述
VPTB[42:39]	[42:39]	WO,0	数据流虚页表基地址。
—	其它位	—	保留。

3.4.11 DC_CTL

DC_CTL为数据Cache的控制寄存器，可读写，用于实现对数据Cache的各种控制，复位时为“0xF”。

表3-35: DC_CTL寄存器域描述

名称	范围	类型	描述	
—	其它	—	保留。	
ECACHE_EN	[18]	RW,0	表示是否核外存在Cache，为“1”时，核心可将核外Cache预取指令、核外Cache刷新指令发出核心，否则当作空指令。	
L2DTB_PERR_EN	[17]	RW,0	二级DTB小页阵列的偶校验使能；L2DTBS增加偶校验；当EBOX查询L2DTBS时，对于该索引的四路的任意一路（可以多路），如果其有效、有偶校验错，且该位使能，则向Rob报自陷，硬件会记录出错的年龄、索引地址和组号，当收到该指令的清空流水线时，硬件会根据索引地址和组号对发生偶校验错的几路置无效。	
F_DTBMISS	[16]	RW,0	强制装填指令处理DTB miss；该位有效时，装填指令访问DTB发生脱靶时，需要进入脱靶处理流程。	
PRB_CTRL	[15]	RW,0	强制关闭SQMB使能；该位有效，当SQMB处于合并状态1024个GCLK时钟周期时，强制关闭SQMB。	
F_DC_SERR	[14]	RW,0	强制产生数据Cache ECC单错。	只对申威411处理器仿真验证有意义，正常使用时应设置为全“0”。
F_DC_MERR	[13]	RW,0	强制产生数据Cache ECC多错。	
F_DT_PAR	[12]	RW,0	强制产生数据Cache标记偶校验错。	
—	[11]	—	保留。	

ATOP_OVF_EN	[10]	RW,0	原子操作指令溢出报错使能。
DCDAT_ERR_EN	[9]	RW,0	数据Cache的数据ECC校验错使能。
PROBE_ERR_EN	[8]	RW,0	查询请求的错误使能。
ACK_ERR_EN	[7]	RW,0	二级Cache管理部件给数据Cache管理部件的响应错使能。
CSR_ERR_EN	[6]	RW,0	CSR写冲突错使能。
EVICTD_ERR_EN	[5]	RW,0	Cache淘汰指令（EVICTDL和EVICTDG）错误检查使能，该位必须设置为1，否则硬件挂死。
DCTAG_PAR_EN	[4]	RW,0	数据Cache的标记偶校验错使能。
SET_EN[3:0]	[3:0]	RW,0xF	数据Cache的组使能，只允许配置为所有组使能或只有一个组被使能，其它组合将导致不可预测的结果。

3.4.12 DC_STAT

DC_STAT为数据Cache的状态寄存器，可读写，写“1”清除。该寄存器用于记录数据Cache管理部件发现的机器检查错状态，这些错都不可纠正，当机器检查错中断使能时，可产生机器检查错中断请求。

表3-36: DC_STAT寄存器域描述

名称	范围	类型	描述
DAT_MERR	[2]	R,W1C	存储器写指令在读出数据Cache中数据进行读修改写时，产生不能被写数据完全覆盖的ECC不可纠正多错，且该指令已退出。在该标志置起时，向软件报非精确的机器检查错，并向维护接口报多错。
PROBE_ERR	[1]	R,W1C	二级Cache管理部件送数据Cache管理部件的查询请求与数据Cache管理部件中记录的状态不一致。在该标志置起时，报非精确的机器检查错。
ACK_ERR	[0]	R,W1C	访存指令产生的一次请求，通过二级Cache管理部件发给存储控制器，从存储控制器返回二级Cache管理部件的响应是错误响应，且该指令未退出。在该标志置起时，对应的指令报精确的机器检查错。
—	其它位	—	保留。

3.4.13 DA_MATCH

DA_MATCH 为数据流地址匹配寄存器，可读写，复位时清“0”。该寄存器用于数据流调试，当数据流访问地址与该寄存器内容匹配时产生数据流故障。

表3-37: DA_MATCH寄存器域描述

名称	范围	类型	描述
—	[63:46]	—	保留。
PA	[45]	RW,0	物理地址标志，为“1”表示物理地址比较。即数据流的物理地址PA[39:0]与ADDR[39:0]进行比较。否则进行虚地址比较，即数据流的虚地址VA[42:0]与ADDR[42:0]进行比较； 注1：具体比较时首先根据DA_MASK[42:0]决定该位是否进行比较； 注2：具体比较时按DA_MATCH_MODE[42:0]指定的类型进行相等比较或不等比较； 注3：对于HM模式的物理地址访问（hw_st/p、hw_ld/p）物理PA指示，无条件按物理地址比较；
EN[1:0]	[44:43]	RW,0	地址匹配使能位，“00 ₂ ”指示禁止地址比较；“01 ₂ ”指示允许读访问地址比较；“10 ₂ ”指示允许写地址比较；“11 ₂ ”指示允许读写地址比较。
ADDR[42:0]	[42:0]	RW,0	与数据流地址进行比较的地址。

3.4.14 DA_MASK

DA_MASK 为数据流地址屏蔽寄存器，可读写。该寄存器用于数据流调试，屏蔽进行数据流地址比较的位数。复位时设置为“0x7FF,FFFF,FFFF”。

表3-38: DA_MASK寄存器域描述

名称	范围	类型	描述
—	[63:43]	—	保留。
MASK[42:0]	[42:0]	RW,全“1”	数据流地址匹配屏蔽位，与DA_MATCH[ADDR]中的每一位对应，为“1”表示该位参与比较。

3.4.15 DA_MATCH_MODE

DA_MATCH_MODE 为数据流地址匹配模式寄存器，可读写。该寄存器用于数据流调试，定义

地址比较的类型 (DA_MASK 具有更高的优先级)。复位时设置为“0x7FF,FFFF,FFFF”。

表3-39: DA_MATCH_MODE寄存器域描述

名称	范围	类型	描述
—	[63:43]	—	保留。
DAMATCH_MODE[42:0]	[42:0]	RW,全“1”	与DA_MATCH[ADDR]中的每一位对应，为“1”表示该位进行相等比较，否则进行不等比较。 注：DA_MASK具有更高的优先级

3.4.16 DVA

DVA 为数据流虚地址寄存器，只读，用于记录数据流地址不对界、DTB 脱靶以及各种数据流故障时的相关数据流地址，DTB_DOUBLE 异常自陷时，该寄存器内容不更新。

表3-39: DVA寄存器域描述

名称	范围	类型	描述
DVA[63:0]	[63:0]	RO	异常时的数据流地址。

一般情况下，该寄存器保留的是数据流虚地址，但以下原因产生数据流故障，记录的是数据流物理地址，只有DVA[39:0]有意义：

- 1) 数据Cache的标记出现偶校验错；
- 2) 数据Cache的数据出现ECC错；
- 3) 数据流地址超出主存容量；
- 4) 访问存储控制器中不存在的I/O寄存器；
- 5) 主存返回的响应数据有ECC多错。

3.4.17 DVA_FORM

DVA_FORM 为数据流虚地址格式寄存器，只读，读出的内容来自 CSR: DVA 和 DS_CTL，目的是便于软件处理 DTB 脱靶。

表3-40: DVA_FORM寄存器域描述

名称	范围	类型	描述
SEXT (DS_CTL[42])	[63:43]	RO	数据流虚页表基地址[42]位的符号扩展。
DS_CTL[42:39]	[42:39]	RO	数据流虚页表基地址。
SEXT(DVA[42])	[38:33]	RO	异常数据流地址虚页号[42]的符号扩展。
DVA[42:13]	[32:3]	RO	异常数据流地址虚页号。

—	[2:0]	—	保留。
---	-------	---	-----

3.5 二级 Cache 管理部件中的 CSR

3.5.1 SC_CTL

SC_CTL为二级Cache控制寄存器，可读写，用于二级Cache控制和错误处理，复位时为“0xFF”。

表3-41： SC_CTL寄存器域描述

名 称	范围	类型	描 述
—	其它位	—	保留。
ICTAG_PAR_EN	[19]	RW,0	核心接收指令副本标记阵列发生偶校验错的报错使能；“1”表示报错使能。
CTL_ERR_EN	[18]	RW,0	SBOX 监测到控制错后登记 SC_STAT[CTL_ERR]并报MCHK的使能位，具体控制错信息可以查看 CSR:SC_STAT[CTL_ERR]。
SYSREQ_ERR_EN	[17]	RW,0	核心接收到外部来的二次请求信息发生校验错（包括单错和多错）的报错使能；“1”表示报错使能。
SYSDAT_ERR_EN	[16]	RW,0	核心接收到外部来的读请求响应数据发生不可纠多错的报错使能；“1”表示报错使能。
SYSACK_ERR_EN	[15]	RW,0	核心接收到外部来的读请求或写请求响应包头发生校验错（包括单错和多错）的报错使能；“1”表示报错使能。
ACK_CHK_EN	[14]	RW,0	收到读请求的控制错响应时的报错使能；“1”表示报错使能。
WR_ERR_EN	[13]	RW,0	收到写请求的控制错响应或非法地址响应时的报错使能；“1”表示报错使能。
IS_FILL_CTL	[12]	RW,0	指令流数据装填控制，为“0”时，指令不引起数据Cache淘汰时装填二级Cache，为“1”时，指令总是不装填二级Cache。
MB_CTL	[11]	RW,0	存储器栏栅控制，为“0”时，将MAF和

			IOWB中所有悬挂请求发送到存储控制器或系统接口，为“1”时，将MAF、IOWB和VAF中所有悬挂请求发送到存储控制器或系统接口，但不包括VAF中的查询响应。
CACHE_ST_ERR_EN	[10]	RW,0	二级Cache状态错检查使能。
SCDAT_ERR_EN	[9]	RW,0	二级Cache数据和数据Cache淘汰数据的ECC校验错使能。
SCTAG_ERR_EN	[8]	RW,0	二级Cache标记的ECC校验错使能。
SET_EN[7:0]	[7:0]	RW,0xFF	二级Cache组使能，对应二级Cache的8个组，只允许全部被使能或只有一个组被使能，其它组合会产生不可预测的结果。

3.5.2 SC_STAT

SC_STAT为二级Cache的状态寄存器，可读写，写“1”清除。该寄存器用于记录二级Cache标记ECC错或数据ECC错，对于ECC单错，如果已纠正错中断使能，则可产生已纠正错中断请求；对于ECC多错等不可纠正错，如果机器检查错中断使能，则可产生机器检查错中断请求。

表3-42: SC_STAT寄存器域描述

名称	范围	类型	描述
—	其它位	—	保留
ICTAG_PAR	[16]	R,W1C	指示指令副本标记阵列发生了偶校验错。
WR_ERR	[15]	R,W1C	指示I/O写请求或者共享不可Cache写请求，或者挤占SCache时产生的被动淘汰请求，收到的响应带控制错或ECC多错标志，该错不可纠正。受SC_CTL[WR_ERR_EN]控制。
CTL_ERR	[14]	R,W1C	分为以下几种情况 1、主核收到的响应与发出的一次请求类型不匹配，包括IO读、Evict类请求、原子操作、置脏请求遇到非法地址响应； 2、主核收到的响应与悬挂缓冲SMAF不匹配； 3、IO写取数请求类型错误；

			<p>4、二次请求或响应与SCache状态不匹配；</p> <p>5、数据流请求或DCache淘汰与SCache状态不匹配。</p> <p>使能位为SC_CTL[CTL_ERR_EN]。</p>
SYSREQ_ERR	[13]	R,W1C	核心接收到外部来的二次请求信息发生校验错（包括单错和多错）。受SC_CTL[SYSREQ_ERR_EN]控制。
SYSDAT_MERR	[12]	R,W1C	核心接收到外部来的读响应数据时，产生ECC不可纠正多错。受SC_CTL[SYSDAT_ERR_EN]控制。
SYSDAT_SERR	[11]	R,W1C	核心接收到外部来的读响应数据时，产生ECC可纠正单错。受SC_CTL[SYSDAT_ERR_EN]控制。
SYSACK_ERR	[10]	R,W1C	核心接收到外部来的读请求或写请求响应包头信息发生校验错（包括单错和多错）。受SC_CTL[SYSACK_ERR_EN]控制。
WR_ADDR_ERR	[9]	R,W1C	指示 I/O 写请求或者共享不可 Cache 写请求，或者挤占 SCache 时产生的被动淘汰请求，收到的非法地址响应，该错不可纠正。受 SC_CTL[WR_ERR_EN]控制。
CACHE_ST_ERR	[8]	R,W1C	指示 Cache 状态与相关请求之间存在不一致性，该错不可纠正。
DVIC_MERR	[7]	R,W1C	指示数据Cache中脏数据被替换或淘汰时，产生ECC不可纠正多错。
DVIC_SERR	[6]	R,W1C	指示数据Cache中脏数据被替换或淘汰时，产生ECC可纠正单错。
DAT_MERR	[3]	R,W1C	指示在读二级Cache的数据存储器时，产生ECC不可纠正多错。
DAT_SERR	[2]	R,W1C	指示在读二级Cache的数据存储器时，产生ECC可纠正单错。
TAG_MERR	[1]	R,W1C	指示在查询二级Cache的标记存储器时，产生ECC不可纠正多错。
TAG_SERR	[0]	R,W1C	指示在查询二级Cache的标记存储器时，产生ECC可纠正单错。

3.5.3 INT_REQ

INT_REQ为核间中断请求寄存器，软件写该寄存器，硬件将发出核间中断0、核间中断1、睡眠中断或者睡眠唤醒中断。为了避免不同类型的中断或不同源和目标之间的核间中断丢失，软件写该寄存器时，需要先读该寄存器，当该寄存器的有效位为“0”时，才执行写操作。复位时，全部清“0”。

表3-43: CORE_INTREQ寄存器域描述

名称	范围	类型	描述
—	其它位	—	保留
Valid	[25]	RW,0	中断请求有效位，当该位为“1”时，表示该中断请求有效，且尚未发送出去。软件必须读到该位为“0”时，才能写INT_REQ发中断，且要求写数据中Valid位为“1”。
IntType[1:0]	[24:23]	RW,0	中断发送类型：为“00 ₂ ”核间中断类型0，为“01 ₂ ”指示核间中断类型1，为“10 ₂ ”指示睡眠中断，为“11 ₂ ”指示睡眠唤醒中断。
Target[15:0]	[15:0]	RW,0	发送中断的目标核心，支持多播中断（多位为“1”）。

3.5.4 CP_CTL

一致性处理控制寄存器（CSR:CP_CTL）控制一致性请求和响应的处理过程。该寄存器为只读属性，只能在芯片初始化复位的配置状态，通过串行扫入修改其内容，除上电复位和冷复位外，其它复位均对该寄存器不起作用。

表3-44: CP_CTL寄存器域描述

名称	范围	类型	描述
—	其它	—	保留
IC_COH	[7]	RO,1	硬件支持ICache一致性控制位。配置为“1”表示核心支持ICache一致性，此时要求核外同时记录SCache和ICache中的副本情况；配置为“0”表示核心不支持ICache一致性，此时核外不需要记录ICache中的副本情况。
ICD	[6]	RO,0	清洁置脏内部确认控制位。可Cache写请求命中Cache状态为清洁独占时，配置为“1”表示核心内部产生置脏成功响应；否则表示核心将根据CTD位的配置，选择向外部系统发送CleanToDirty或ShareToDirty请求。
MB	[5]	RO, 0	存储器栏栅请求控制位。配置为“1”表示核心会发出存储器栏栅请求，由外部系统确认存储器栏栅是否完成；否则表示

			核心不会发出存储器栏栅请求，内部确认存储器栏栅是否完成，此时要求外部系统在响应中携带请求完成标志。
EGB	[4]	RO, 1	全局Cache淘汰控制位。配置为“1”表示核心会发出全局Cache淘汰请求，要求外部系统支持全局的Cache内容淘汰（通过二次请求实现）；否则表示核心不会发出此请求，使用EVICTDG指令将产生非法指令异常。
ELB	[3]	RO, 1	本地Cache淘汰控制位。配置为“1”表示核心会发出本地Cache淘汰请求，要求外部系统支持本核心Cache内容的淘汰（通过二次请求实现）；否则表示核心不会发出此请求，使用EVICTDL和FLUSHD指令将产生非法指令异常。
CTD	[2]	RO, 1	清洁置脏控制位。可Cache写请求命中Cache状态为清洁独占时，配置为“1”表示核心将发出CleanToDirty请求，否则表示核心将发出ShareToDirty请求。
CVB	[1]	RO, 0	清洁淘汰控制位。Cache装填引起有效Cache清洁内容淘汰时，配置为“1”表示核心需要通知外部系统；否则表示核心不需要通知外部系统。
VB_ORDER	[0]	RO, 0	淘汰请求顺序控制位。Cache装填引起有效Cache内容淘汰时，配置为“1”表示核心先发出WrVictimBlk或CleanVictimBlk请求，后发出ReadBlkI、ReadBlk、或ReadBlkModify请求；否则表示核心先发出ReadBlkICVic、ReadBlkIDVic、ReadBlkCVic、ReadBlkDVic、ReadBlkModifyCVic或ReadBlkModifyDVic请求，随后发出WrVictimBlk或CleanVictimBlk。

3.5.5 MERGE_OVTIME

MERGE_OVTIME为I/O空间写指令和存储器不可Cache空间写指令的合并窗口超时配置，可读写，复位为“0x100”。

表3-45: MERGE_OVTIME寄存器域描述

名称	范围	类型	描述
—	其它位	—	保留。
OVTIME	[15: 0]	RW,0x100	写合并窗口超时配置寄存器，为IO写和存储器不可Cache空间写共用的配置。当合并等待计数器等于本字段时，硬件自动关闭合并窗口，形成IO写请求或者不可Cache写请求，同时对超时计数器清

			0。 特殊的，当本寄存器等于0时，表示所有IO写和不可Cache写请求不可合并。
--	--	--	---

3.6 整数部件中的 CSR

3.6.1 IVA

IVA 为指令虚地址寄存器，只读，当跳转指令的目标指令地址出现符号扩展错时，该寄存器记录出错的跳转目标指令虚地址；当整数转移指令的目标地址匹配成功时，该寄存器记录匹配成功的转移目标指令虚地址。

表3-46: IVA寄存器域描述

名称	范围	类型	描述
IVA[63:0]	[63:0]	RO	指令虚地址

3.6.2 TC

TC 为计时器，可读写，作为周期计数器。当计数使能时，每个时钟周期此计数器“+1”。复位时设置为全“0”。

表3-47: TC寄存器域描述

名称	范围	类型	描述
TC[63:0]	[63:0]	RW,0	计时器的计数值。

3.6.3 TC_CTL

TC_CTL 为计时器控制寄存器，可读写，用于控制 TC 计数器的计数使能。复位时设置为“0”。

表 3-48: TC_CTL 寄存器域的描述

名称	范围	类型	描述
—	[63:1]	—	保留。
TC_ENA	[0]	RW,0	计时器计数使能，为“1”时计时器开始计数。

3.6.4 ICR

ICR 为整数控制寄存器，可读写，控制是否产生整数溢出自陷，复位时清“0”。

表 3-49: ICR 寄存器域的描述

名称	范围	类型	描述
—	[63:1]	—	保留。
FD	[0]	RW, 0	整数溢出屏蔽位。为“1”时，整数运算指令和原子操作指令不产生任何算术自陷。

3.6.5 CID

CID 为核心识别码寄存器。

表 3-50: CID 寄存器域的描述

名称	范围	类型	描述
CID[63:8]	[63:8]	RW	可以写入全“0”，也可以写入系统中多核 CPU 的芯片编号。
CID[3:0]	[3:0]	RO	本核心在芯片内部的编号。
—	其它位	—	保留。

3.6.6 IDA_MATCH

IDA_MATCH 为指令流目标地址匹配寄存器，可读写，复位时清“0”。该寄存器用于指令流调试，当整数转移指令的目标地址与该寄存器内容匹配时，产生指令流故障。

表3-51: IDA_MATCH寄存器域描述

名称	范围	类型	描述
EN	[51]	RW,0	匹配使能信号，为“1”表示匹配使能。
ADDR[42:0]	[42:0]	RW,0	需要匹配的整数转移目标指令虚地址，其中[1:0]位指定需要匹配的处理器模式。
—	其它位	—	保留。

3.6.7 IDA_MASK

IDA_MASK 为指令流目标地址屏蔽寄存器，可读写，复位时为“0x7FF,FFFF,FFFF”。该寄存器用于指令流调试，屏蔽进行整数转移目标地址比较的位数。

表3-52: DS_CTL寄存器域描述

名称	范围	类型	描述
—	[63:43]	—	保留。

MASK[42:0]	[42:0]	RW,全“1”	整数转移目标地址匹配屏蔽位，对应IDA_MATCH[ADDR]每位，为“1”表示该位参与相等比较。
------------	--------	---------	---

3.7 浮点运算部件中的 FPCR

3.7.1 FPCR 描述

浮点运算部件内部设置了一个专门控制浮点指令执行和记录浮点指令执行状态的浮点控制寄存器 FPCR, 该寄存器属于特殊的 CSR, 只能通过非特权指令 RFPCR、WFPCR、SETFPEC0、SETFPEC1、SETFPEC2、SETFPEC3 进行读写访问, 表 3-51 为申威 411 处理器实现的 FPCR 的内容。

表 3-52: FPCR 寄存器域描述

寄存器位	标识	含义
63	SUM	总算术异常标志: [57:52]、[41:36]、[25:20]、[9:4]等位中存在“1”, 则该位为“1”。
62	INED	非精确结果自陷屏蔽: 屏蔽非精确结果自陷并将IEEE标准指定的结果写入目标寄存器。
61	UNFD	下溢自陷屏蔽: 屏蔽下溢自陷, 该位设置为“1”且UNDZ位设置为“1”才能实现下溢自陷屏蔽。
60	UNDZ	下溢为“0”: 该位与UNFD位共同控制下溢自陷屏蔽。不论该位设置为“0”还是“1”, 下溢时写入目标寄存器中的结果总是真零(64位全“0”)。
[59:58]	DYN	动态舍入模式, 指示指令的具体舍入模式, 具体定义如下: 00 ₂ : 向“0”舍入 01 ₂ : 向负无穷大舍入 10 ₂ : 就近舍入 11 ₂ : 向正无穷大舍入
[57:52]位为非SIMD浮点运算指令或者浮点SIMD运算指令中浮点向量第0个元素运算产生的六种浮点算术异常。		
57	OVI0	整数溢出: 浮点转换为整数的操作产生整数溢出。
56	INE0	非精确结果: 浮点算术运算或转换操作的结果与精确结果不同。
55	UNF0	下溢: 浮点算术运算或转换操作结果的指数产生下溢。
54	OVF0	上溢: 浮点算术运算或转换操作结果的指数产生上溢。
53	DZE0	除数为“0”: 浮点除法运算的除数为“0”。

52	INV0	无效操作：浮点算术、转换或比较运算中，一个或多个操作数是非有限数，或者对具体操作而言，操作数是非法的。
51	OVFD	上溢自陷屏蔽：禁止上溢自陷，并根据中间结果和舍入模式确定的结果写入目标寄存器。
50	DZED	除数为“0”自陷屏蔽：禁止除数为“0”自陷，并根据中间结果和舍入模式确定的结果写入目标寄存器。
49	INVD	无效操作自陷屏蔽：禁止无效操作自陷，并根据中间结果和舍入模式确定的结果写入目标寄存器。
48	DNZ	非规格化数为“0”：将所有非规格化操作数当作符号相同的“0”值。
[47:42]	RAZ/IGN	保留位，读出为“0”，写时忽略。
[41:36]位为浮点SIMD运算指令中浮点向量第1个元素运算产生的六种浮点算术异常。		
41	“0”	固定为“0”。
40	INE1	非精确结果：浮点算术运算或转换操作的结果与精确结果不同。
39	UNF1	下溢：浮点算术运算或转换操作结果的指数产生下溢。
38	OVF1	上溢：浮点算术运算或转换操作结果的指数产生上溢。
37	DZE1	除数为“0”：浮点除法运算的除数为“0”。
36	INV1	无效操作：浮点算术、转换或比较运算中，一个或多个操作数是非有限数，或者对具体操作而言，操作数是非法的。
[35:26]	RAZ/IGN	保留位，读出为“0”，写时忽略。
[25:20]位为浮点SIMD运算指令中浮点向量第2个元素运算产生的六种浮点算术异常。		
25	“0”	
24	INE2	非精确结果：浮点算术运算或转换操作的结果与精确结果不同。
23	UNF2	下溢：浮点算术运算或转换操作结果的指数产生下溢。
22	OVF2	上溢：浮点算术运算或转换操作结果的指数产生上溢。
21	DZE2	除数为“0”：浮点除法运算的除数为“0”。
20	INV2	无效操作：浮点算术、转换或比较运算中，一个或多个操作数是非有限数，或者对具体操作而言，操作数是非法的。
[19:10]	RAZ/IGN	保留位，读出为“0”，写时忽略。
[9:4]位为浮点SIMD运算指令中浮点向量第3个元素进行运算产生的六种浮点算术异常。		
9	“0”	固定为“0”。
8	INE3	非精确结果：浮点算术运算或转换操作的结果与精确结果不同。
7	UNF3	下溢：浮点算术运算或转换操作结果的指数产生下溢。

6	OVF3	上溢：浮点算术运算或转换操作结果的指数产生上溢。
5	DZE3	除数为“0”：浮点除法运算的除数为“0”。
4	INV3	无效操作：浮点算术、转换或比较运算中，一个或多个操作数是非有限数，或者对具体操作而言，操作数是非法的。
[3]	RAZ/IGN	保留位，读出为“0”，写时忽略。
[2]	EXC_CTL_WEN	保留位，读出为“0”，写时忽略。
[1:0]	EXC_CTL	控制浮点算术自陷（含浮点部件的整数溢出异常自陷）屏蔽模式，选择使用FPCR其它控制位的方式： “00 ₂ ”：允许所有浮点算术异常自陷且受FPCR中控制位的控制； “01 ₂ ”：允许除不精确结果之外的所有浮点算术异常自陷且受FPCR中控制位的控制； “10 ₂ ”：允许除不精确结果之外的所有浮点算术异常自陷且不受FPCR中控制位的控制； “11 ₂ ”：允许除浮点下溢、不精确结果和整数溢出之外的所有浮点算术异常自陷且不受FPCR中控制位的控制。

说明：

1) 在申威 411 处理器中，不支持非规格化数自陷屏蔽（DNOD: Denormal Operand Exception Disable），当设置 FPCR[48]为“1”时可屏蔽非规格化数自陷；

2) 在申威 411 处理器中，因浮点 SIMD 运算指令不包含转换指令，因此浮点 SIMD 运算不会产生整数溢出异常；

3) FPCR 中浮点算术异常标志的设置与自陷是否屏蔽无关，即使某些异常自陷被屏蔽，一旦产生浮点算术异常，仍要在 FPCR 中记录相应的异常标志，这些标志位记录为“1”后，只有在软件将“0”写入这些位，才能清除对应的异常标志位；

4) 读 RFPCR 由 FPCR 指令完成，写 FPCR[63:2]由 WFPCR 指令完成，写 FPCR[1:0]则由 SETFPEC0、SETFPEC1、SETFPEC2、SETFPEC3 指令完成，分别将 FPCR[1:0]更新为 2'b00、2'b01、2'b10 和 2'b11。由于 FPCR 对浮点指令的影响，写 FPCR 时，必须保证其不影响写 FPCR 指令之前的浮点指令，并对写 FPCR 指令之后的浮点指令要确保产生作用。读 FPCR 时，为了回收所有异常情况，也必须等读 FPCR 指令之前的浮点指令都完成，包括可能的异常处理，在读 FPCR 指令完成之前，保证不执行新的浮点指令；

5) 处理器初始化后，在使用浮点指令之前，必须对 FPCR 进行初始化。进程切换时，必须对 FPCR 进行保留与恢复；

6) RFPCR、WFPCR、SETFPEC0、SETFPEC1、SETFPEC2 和 SETFPEC3 指令实现了 FPCR 与浮点寄存器之间的数据传送，因 FLDS 和 FSTS 指令会对浮点寄存器的[61:59]位进行处理，

因此不能用这两条指令来处理 FPCR 的值。

3.7.2 FPCR 访问机制

FPCR 只能通过 RFPCR、WFPCR、SETFPEC0、SETFPEC1、SETFPEC2 和 SETFPEC3 指令进行显式读写操作，执行浮点指令时，硬件执行隐式读操作，根据 FPCR 中的控制位判断浮点异常处理方式和浮点运算结果，如果发生浮点算术异常，硬件自动对 FPCR 进行隐式写操作。对 FPCR 访问顺序有以下几种情况：

1) 对于 FPCR[62:58]、FPCR[51:48]、FPCR[2:0]，只存在显式读、显式写、隐式读操作，因此不存在同时隐式读和隐式写操作；

2) 对于其它位，除保留位外，只存在显式读、显式写、隐式写操作，因此也不存在同时隐式读和隐式写操作。

另外，指令部件设置一位专用的 FPCR 记分板来保证读写之间的顺序，具体如下：

1) WFPCR、SETFPEC0、SETFPEC1、SETFPEC2 和 SETFPEC3 指令准备发射时，必须要求 FPCR 记分板被清除；该指令发射以后，立即置 FPCR 记分板；该指令退出时，清除 FPCR 记分板；

2) RFPCR 指令准备发射时，必须要求 FPCR 记分板被清除，且前面所有的指令都已正常退出；

3) 浮点运算指令准备发射时，必须要求 FPCR 记分板被清除。

同时，浮点运算部件也提供特殊的写机制来协助保证读写之间的顺序。

1) WFPCR、SETFPEC0、SETFPEC1、SETFPEC2 和 SETFPEC3 指令退出时，根据写的值更新 FPCR 中的内容；

2) 浮点运算指令退出时，如果发生浮点算术异常，无论自陷是否屏蔽，将该指令的异常信息登记到 FPCR 中。

4 存储管理和存储结构

4.1 存储空间

- 1) 申威 411 处理器支持的虚拟地址空间为 43 位 (VA[42:0]), 物理地址空间为 40 位 (PA[39:0]);
- 2) 物理地址的最高位 PA[39]用于区分存储器空间 (PA[39]位为“0”)和 I/O 空间 (PA[39]位为“1”)。指令流总是访问存储器空间, 即固定指令流访问的物理地址 PA[39]位为“0”。

4.1.1 存储器空间

- 1) 数据流和指令流都能对存储器空间进行访问, 其中数据流访问的指令包括所有 Load 类型和 Store 类型的访存指令, 另外还包括淘汰、刷新和数据装填等 Cache 控制指令;
- 2) 申威 411 处理器支持两类访存指令, 一类是普通访存指令, 根据所访问的存储器空间属性来决定数据是否进入核心 Cache; 另一类是不可 Cache 访存指令, 所访问的数据总是不进入核心 Cache;
- 3) 对于普通访存指令, 物理地址 PA[38]位作为可 Cache 属性标志位, 为“0”表示可 Cache, 为“1”表示不可 Cache; 对于不可 Cache 访存指令, 物理地址 PA[38]无意义;
- 4) PA[37:0]位作为真正的存储器空间物理地址, 用于访问核外存储器空间, 核外可进行进一步的存储空间划分。

由此推论:

- 1) PA[39]为“0”, PA[37:0]相同, PA[38]不同时, 认为是同一个物理地址;
- 2) 对一个 PA[37:0]确定的存储器空间物理地址, 申威 411 处理器可以通过三种方式访问该地址:
普通访存指令进行可 Cache 访问 (PA[38]拼接“0”)、普通访存指令进行不可 Cache 访问 (PA[38]拼接“1”)、不可 Cache 访存指令进行不可 Cache 访问 (PA[38]任意), 后两种访问方式的硬件处理流程相同。

4.1.1.1 可 Cache 特性

1) 数据流访问特性

数据流进行共享可 Cache 读写访问时, 此时读访问数据总是进入数据 Cache 和二级 Cache, 写访问数据总是先进入数据 Cache 并在数据 Cache 中完成写操作, 从数据 Cache 淘汰时才写入二级 Cache。申威 411 处理器保证数据 Cache 的内容为二级 Cache 的子集, 二级 Cache 的标记中包含特殊的状态, 用来指示最新数据在数据 Cache 中还是在二级 Cache 中。申威 411 处理器基于二级 Cache

来保证数据流对可 Cache 空间访问的一致性。为提高性能，装填指令可以将可 Cache 空间上的数据提前预取到数据 Cache、二级 Cache、或核外三级 Cache 中。

2) 指令流访问特性

指令流进行共享可 Cache 读访问时，读访问数据进入指令 Cache，可进入也可不进入二级 Cache，不保证指令 Cache 内容为二级 Cache 的子集。当核心选择支持指令 Cache 一致性的配置时，要求核外逻辑同时记录指令 Cache 内容为二级 Cache 中的副本状态；当核心选择不支持指令 Cache 一致性的配置时，要求软件来解决指令 Cache 一致性问题。为提高性能，申威 411 处理器支持指令流预取，可以将可 Cache 空间上的指令提前预取到指令 Cache 中。

3) 访问合并原则

处理对可 Cache 存储器空间访问时，新产生的请求与之前产生且未完成处理的请求进行地址比较，如果满足条件则可将请求合并。具体规则见表 4-1。

表 4-1: 共享可 Cache 空间访问的合并规则

		已产生且未完成处理的请求					
		LDx	STx	FETCHD	FLUSHD	EVICTD	指令流
新产生的请求	LDx	合并	合并	合并	—	—	—
	STx	合并	合并	合并	—	—	—
	FETCHD	合并	合并	合并	—	—	—
	FLUSHD	—	—	—	—	—	—
	EVICTD	—	—	—	—	—	—
	指令流	—	—	—	—	—	合并

说明：

- 1) 已有的请求相关的指令与新产生的请求相关的指令是按程序顺序定义的；
- 2) 表中指示“合并”的位置，表示两个请求可以合并；
- 3) 表中指示“—”的位置，表示不存在或不允许请求合并。

4.1.1.2 不可 Cache 特性

1) 数据流访问特性

数据流对存储器空间进行共享不可 Cache 读写访问时，读访问数据不进入数据 Cache 和二级 Cache，申威 411 处理器保证数据流访问不可 Cache 空间时，总能获得最新数据副本。如果淘汰指令或预取指令的访问地址是不可 Cache 存储器空间，则当空指令处理。

2) 指令流访问特性

指令流对其它核组存储器空间只能进行共享不可 Cache 读访问，读访问数据进入指令 Cache，但不进入二级 Cache，申威 411 处理器保证指令流访问不可 Cache 空间时，总是获得最新数据副本。

3) 访问合并原则

数据流访问不可 Cache 存储器空间时，也要与已产生且未完成的同类请求进行地址比较，判断是否可以合并。

对于读访问，如果新产生的请求地址与已产生的请求地址属于同一个 Cache 行、地址不重叠且该地址对应的合并窗口未关闭，则新请求可以合并到已有请求中。

对于写访问，表 4-2 给出了共享不可 Cache 空间 Store 指令的合并原则，其中行代表新放入合并缓冲的 Store 类指令，列代表已在合并缓冲中 Store 类指令的数据长度。

表 4-2: Store 指令访问共享不可 Cache 空间合并规则

新指令 合并缓冲	字节 指令	半字指 令	字/单精 度指令	长字/双精 度指令	32 字节 SIMD 指令	64 字节 SIMD 指令
字节	合并成 16 字节	—	—	—	—	—
半字	—	合并成 32 字节	—	—	—	—
字 单精度浮点	—	—	合并成 64 字节	—	—	—
长字 双精度浮点	—	—	—	合并成 128 字节	—	—
32 字节 SIMD 向量	—	—	—	—	合并成 128 字 节	—
64 字节 SIMD 向量	—	—	—	—	—	合并成 128 字节

对表 4-2 的规则总结如下：

- 1) 不同数据长度的 Store 指令不允许合并；
- 2) 按程序顺序，允许前后若干条访问数据长度为字节的 Store 指令在自然对界的 16 字节范围内进行合并；
- 3) 按程序顺序，允许前后若干条访问数据长度为半字的 Store 指令在自然对界的 32 字节范围内进行合并；
- 4) 按程序顺序，允许前后若干条访问数据长度为字整数或单精度浮点的 Store 指令在自然对界的 64 字节范围内进行合并；
- 5) 按程序顺序，允许前后若干条访问数据长度为长字整数或双精度浮点的 Store 指令在自然对界的 128 字节范围内进行合并；
- 6) 按程序顺序，允许前后若干条访问数据长度为单精度浮点向量的 Store 指令在自然对界的 128 字节范围内进行合并；

7) 按程序顺序, 允许前后若干条访问数据长度为字整数向量或双精度浮点向量的 Store 指令在自然对界的 128 字节范围内进行合并;

8) 按程序顺序, 允许前后若干条不对界字向量或单精度浮点向量 Store 指令在自然对界的 64 字节范围内进行合并;

9) 按程序顺序, 允许前后若干条不对界双精度浮点向量 Store 指令在自然对界的 128 字节范围内进行合并。

4.1.1.3 访问顺序

申威 411 处理器可以通过 MEMB 指令保证存储器访问的顺序性, 在没有 MEMB 指令时, 不同访存指令之间也要保持一种顺序关系, 具体见表 4-3。

表 4-3: 存储器空间访问顺序 (假定 A 为 PA[37:0]的取值)

前一条指令 后一条指令	Store {0, A}	Store {1, A}	NC_store {x, A}	Load {0, A}	Load {1, A}	NC_Load {x, A}
Store {0, A}	保序	不保序	不保序	保序	保序	保序
Store {1, A}	保序	保序	保序	保序	保序	保序
NC_store {x, A}	保序	保序	保序	保序	保序	保序
Load {0, A}	保序	不保序	不保序	保序	保序	保序
Load {1, A}	保序	不保序	不保序	保序	保序	保序
NC_Load {x, A}	保序	不保序	不保序	保序	保序	保序

说明:

1) 上述表格主要描述普通访存指令访问可 Cache 空间和不可 Cache 空间, 以及不可 Cache 访存指令访问 PA[37:0]相同的物理地址时, 相互之间的硬件执行顺序;

2) 上述访问顺序都是就单个核心内部的程序顺序而言的, 不同核心间的访问顺序必须由软件通过同步机制来保证;

3) 针对表中不保序的地方, 则需要软件增加 MEMB 指令来强制保序;

4) 一般情况下, 不建议软件使用不可 Cache 写类指令, 也不要对 PA[38]为“1”的空间执行写操作, 如果要用, 建议先加一个 MEMB, 然后执行批量不可 Cache 写操作, 最后再加一个 MEMB。

4.1.2 I/O 空间

4.1.2.1 不可 Cache 特性

指令流不产生对 I/O 空间的访问。数据流对 I/O 空间只能执行不可 Cache 访问，不查询数据 Cache 和二级 Cache，也不会引起数据 Cache 和二级 Cache 装填。I/O 空间访问的指令包括所有 Load 类型的指令和 Store 类型的指令，但不包括淘汰、刷新和数据装填等 Cache 控制指令，这些指令访问的数据流地址为 I/O 空间时，都作为空指令。

4.1.2.2 合并特性

I/O 读指令不允许合并，I/O 写指令允许合并，但合并是有限制的，主要限制有：

- 1) 字节或半字的 I/O 写指令不允许合并，不同数据长度的 I/O 写指令不允许合并；
- 2) 按程序顺序，如果前后若干条字 I/O 写指令产生的数据流地址是升序且不重叠（不一定连续），则允许这些指令在自然对界的 64 字节范围内进行合并；
- 3) 按程序顺序，如果前后若干条长字 I/O 写指令产生的数据流地址是升序且不重叠（不一定连续），则允许这些指令在自然对界的 128 字节范围内进行合并；
- 4) I/O 写地址达到可合并地址的最高端时（即无法再合并以后的 I/O 写指令），关闭合并窗口；
- 5) 遇到 MEMB 指令、I/O 读指令或不能合并的 I/O 写指令时，关闭合并窗口；
- 6) 等待合并超时，关闭合并窗口。

关闭合并窗口后，即可将合并后的 I/O 写请求发出核心。

4.1.2.3 访问顺序

申威 411 处理器的核心可以通过 MEMB 指令保证 I/O 空间访问的顺序性，没有 MEMB 指令时，不同访问指令之间，也保持一种顺序关系，具体见表 4-4。

表 4-4: I/O 空间访问顺序（假定 X 与 Y 是不同的地址）

程序中较前面的访存指令	程序中较后面的访存指令	访问顺序	说明
地址 X 的 Load 访问	地址 X 的 Load 访问	保证程序顺序	申威 411 处理器向外部发出访问请求的顺序与程序中指令的顺序一致。
地址 X 的 Load 访问	地址 Y 的 Load 访问	有条件保证程序顺序	
地址 X 的 Store 访问	地址 X 的 Store 访问	保证程序顺序	
地址 X 的 Store 访问	地址 Y 的 Store 访问	有条件保证程序顺序	
地址 X 的 Load 访问	地址 X 的 Store 访问	保证程序顺序	
地址 X 的 Load 访问	地址 Y 的 Store 访问	有条件保证程序顺序	

地址 X 的 Store 访问	地址 X 的 Load 访问	保证程序顺序	
地址 X 的 Store 访问	地址 Y 的 Load 访问	有条件保证程序顺序	

说明:

1) “有条件保证程序顺序”是指 I/O 空间访问物理地址[38:36]位相同的指令之间才保证程序顺序;

2) 上述访问顺序都是就单个核心内部的程序顺序而言的,不同核心间的访问顺序必须由软件通过同步机制来保证。

4.2 Cache 结构

申威 411 处理器包含两级 Cache,一级 Cache 将指令和数据分开,分别为指令 Cache (ICache) 和数据 Cache (DCache),二级 Cache 为指令和数据共用,称之为 SCache。

4.2.1 指令 Cache

申威 411 处理器中的指令 Cache 容量为 32KB,采用 4 路组相联结构、虚地址索引和虚地址标记,Cache 行大小为 128 字节。每个 Cache 行的内容包括:

- 1) 1120 位数据,包括 32 条指令以及这些指令的预译码信息,每条指令有 3 位预译码信息;
- 2) 32 位指令偶校验位,每位对应 1 条指令以及该指令的预译码信息;
- 3) 30 位虚地址标记 (TAG),即虚地址[42:13]位;
- 4) 8 位用户进程号 UPN;
- 5) 1 位系统空间标志 KS;
- 6) 2 位处理器模式位 PM[1:0];
- 7) 1 位有效位;
- 8) 1 位标记偶校验位,包括 TAG、UPN、PM、KS 在内的总校验位。

指令 Cache 采用轮转淘汰策略,对应每个 Cache 行,有 2 位淘汰指针,指示下次进行装填的组号。指令 Cache 的数据存储器 and 地址标记存储器 (ITAG) 都采用偶校验,出现偶校验错时,硬件自动按出错的索引号刷新 4 路指令 Cache,然后从二级 Cache 或存储器中重新取指令。软件写 CSR: IC_FLUSH 时,也可将整个指令 Cache 刷新。

指令流水线取指令不命中指令 Cache 时,通过 ITB,将虚地址转换为物理地址,再访问二级 Cache 或存储器,同时可对后继指令进行预取。返回的指令装填指令 Cache 时,PM 位为引起指令 Cache 装填的指令 PC[1:0],具体处理如下:

- 1) 如果在 HM 模式下装填 ICache,KS 位固定为“1”,UPN 来自 CSR:UPCR[UPN];
- 2) 如果在 VM 模式下装填,且虚地址[42]为“0”,KS 位固定为“1”,UPN 来自 CSR:UPCR[UPN];
- 3) 如果在 VM 模式下装填且虚地址[42]为“1”,或者在 KM 和 UM 下,则 KS 和 UPN 都来自

命中的 ITB 条目。

4.2.2 数据 Cache

申威 411 处理器中的数据 Cache 容量为 32KB，采用 4 路组相联结构、物理地址索引和物理地址标记，并采用回写、读写都可装填 Cache 的策略，Cache 行大小为 128 字节。每个 Cache 行的内容包括：

- 1) 1024 位数据；
- 2) 128 位 ECC 校验码，每 64 位数据对应 8 位 ECC 校验码；
- 3) 25 位物理地址标记，即物理地址[37:13]位；
- 4) 3 位状态位，即有效位（V）、可写位（W）和修改位（M）；
- 5) 1 位标记偶校验位，包括物理地址标记和状态位在内的总校验位。

数据 Cache 采用 FIFO 淘汰策略，对应每个 Cache 行，有 2 位淘汰指针，指示下次进行装填的组号。淘汰指令、数据装填指令影响淘汰指针的修改，具体参考《申威 411 处理器结构手册》。

数据 Cache 的数据存储器采用 ECC 校验，可以纠正单错，检测双错和部分多错。数据 Cache 的标记存储器（DTAG）采用偶校验，大部分偶校验错可以通过数量流故障自陷和对应的特权程序进行纠错。

硬件不支持对数据 Cache 的自动刷新，但保证数据 Cache 中数据的一致性。

4.2.3 二级 Cache

申威 411 处理器中的 SCache 容量为 512KB，采用 8 路组相联结构、物理地址索引和物理地址标记，并采用回写、读写都可装填 Cache 的策略，Cache 行大小为 128B。SCache 既可以存放数据，也可以存放指令。每个 Cache 行内容包括：

- 1) 1024 位数据；
- 2) 128 位 ECC 校验码，每 64 位数据对应 8 位 ECC 校验码；
- 3) 22 位物理地址标记，即物理地址[37:16]位；
- 4) 3 位状态位，即数据有效位（SV）、数据 Cache 副本有效位（DV）和可写位（W）；
- 5) 2 位数据 Cache 组号，指示该 Cache 行在四路联想的数据 Cache 中的组号；
- 6) 6 位标记 ECC 校验位，包括物理地址标记、状态位和数据 Cache 组号在内的总校验位。

SCache 采用 FIFO 淘汰策略，对应每个 Cache 行，有 3 位淘汰指针，用来指示下次进行装填的组号。淘汰指令、数据装填指令不影响淘汰指针的修改方法。

SCache 的数据存储器和标记存储器（STAG）都采用 ECC 校验，可以纠正单错，检测双错和部分多错。

硬件不支持对 SCache 的自动刷新，但保证 SCache 中数据的一致性，且保持数据 Cache 为 SCache

的子集,最新数据副本可能同时在数据 Cache 和 SCache 中,也可能只在数据 Cache 中,或只在 SCache 中。

4.2.4 指令副本标记

为支持指令 Cache 的一致性,核心设置了一级指令 Cache 的副本标记 ICTag 存储器 (ICTag),采用物理地址索引和标记,4 路组相联结构,且与指令指令 Cache 中的副本记录一一对应。每个 Cache 行对应的内容包括:

- 1) 1 位有效位;
- 2) 25 位物理地址标记 (TAG), 即物理地址[37:13]位;
- 3) 1 位偶校验位, 只包括物理地址标记的校验位。

软件指定对指令 Cache 的刷新,以及硬件因偶校验引起的 ICache 自动刷新都将引起 ICTag 存储器刷新。

4.3 地址转换缓冲

申威 411 处理器设置了虚实地址转换缓冲 (TLB) 来实现虚实地址转换和存储保护,指令流地址转换缓冲 (ITB) 和数据流地址转换缓冲 (DTB) 分开,有利于提高效率。

对特殊存储空间的指令流或数据流地址,还可以直接将虚地址作为物理地址来访问,将这种直接的地址映射模式称之为“超页”模式。在 ITB 和 DTB 中,设置有存储保护控制和大页机制,以实现存储空间保护和提高 TLB 的命中率。申威 411 处理器还支持系统空间标志 (KS)、虚拟机号 (VPN) 和用户进程号 (UPN),实现对用户进程和系统进程的分别处理。

4.3.1 指令流地址转换缓冲 (ITB)

指令流地址转换缓冲 (ITB) 由 32 个条目组成,全相联结构,采用轮转 (Round-Robin) 替换策略。由于指令 Cache 采用虚地址索引和虚地址标记,因此,仅当指令 Cache 不命中时才访问 ITB。ITB 的主要内容如下:

- 1) 30 位虚地址页号, 即虚地址[42:13]位;
- 2) 26 位物理页面号, 即物理地址[38:13]位;
- 3) 2 位页面粒度 GH[1:0], 分别指示该 ITB 条目对应 1、32、1024 或 32768 个连续的 8KB 页;
- 4) 3 位访问控制位 (UEE、KEE、VEE), 对应用户模式、内核模式、虚拟模式的可执行控制位;
- 5) 2 位虚拟机号 VPN;
- 6) 8 位用户进程号 UPN;

- 7) 1 位系统空间标志 KS;
- 8) 1 位有效位。

申威 411 处理器指令流只允许访问存储器空间，因此不管写 CSR: ITB_PTE[39]位的内容，指令流产生的存储访问请求的物理地址[39]位固定为“0”。

ITB 装填策略是，首先选择无效的条目，若没有无效的条目，则按轮转指针来选择装填的条目。申威 411 处理器在 ITB 装填时，先要进行查询，如果要装填的条目因为粒度不同而命中多个条目，则将新条目写入第一个命中的条目中，并将其它条目清除，以保证后续访问最多只命中一个 ITB 条目。

ITB 的刷新方式有五种，都是通过写以下相关 CSR 来实现的：

- 1) 对指定虚页面的 ITB 条目刷新：写 CSR: ITB_IS 实现；
- 2) 对指定用户进程号的 ITB 条目刷新：写 CSR: ITB_IU 实现；
- 3) 对指定虚拟机号的全部用户进程的 ITB 条目刷新：写 CSR: ITB_IVP 实现；
- 4) 对指定虚拟机号的 ITB 条目刷新：写 CSR: ITB_IV 实现；
- 5) 对所有 ITB 条目刷新：写 CSR: ITB_IA 实现。

当操作系统需要更改指令空间虚实地址对应关系，在刷新 ITB 的同时，必须刷新指令 Cache，以保持指令 Cache 与 ITB 的一致性。

4.3.2 数据流地址转换缓冲 (DTB)

数据流地址转换缓冲 (DTB) 由两级地址转化缓冲组成，一级数据地址转化缓冲 (L1DTB) 由 64 个条目组成，全相联结构，采用轮转替换策略。每个 DTB 条目的内容如下：

- 1) 30 位虚地址页号，即虚地址[42:13]位；
- 2) 27 位物理页面号，即物理地址[39:13]位；
- 3) 2 位页面粒度 GH[1:0]，分别指示该 DTB 条目对应 1、32、1024 或 32768 个连续的 8KB 页；
- 4) 4 位处理器模式读控制位 URE、KRE、VRE 和 HRE，指示该 DTB 条目对应的页中数据是否允许在指定的处理器模式下进行读访问；
- 5) 4 位处理器模式写控制位 UWE、KWE、VWE 和 HWE，指示该 DTB 条目对应的页中数据是否允许在指定的处理器模式下进行写访问；
- 6) 2 位访问权限控制位 FOR (读故障) 和 FOW (写故障)，指示对该 DTB 条目对应的页中数据进行读或写访问时，将产生故障；
- 7) 2 位虚拟机号 VPN；
- 8) 8 位用户进程号 UPN；
- 9) 1 位系统空间标志 KS；
- 10) 1 位有效位。

L1DTB 装填策略是，首先选择无效的条目，若没有无效的条目，则按轮转淘汰指针来选择装填

的条目。

二级数据地址转化缓冲 (L2DTB) 由两个地址表组成, 一个是 512 条目的小页表 (L2SDTB), 一个是 32 条目的大页表 (L2BDTB), 两个表中的条目存放内容都与 L1DTB 相同。L2SDTB 为四路组相联结构, 只存放 8KB 的页面对应的地址映射关系; L2BDTB 是全相联结构, 用于存放大页 (32、1024 或 32768 个连续的 8KB 页) 对应的地址映射关系。

硬件保证一级 DTB 和二级 DTB 是互斥关系, 当不命中一级 DTB, 而命中二级 DTB 时, 硬件自动将二级 DTB 中命中的条目读出, 装填到一级 DTB 中, 如果一级 DTB 中有条目淘汰, 则将淘汰的条目写回二级 DTB; 如果两级 DTB 都不命中, 则需要软件装填, 此时硬件将装填一级 DTB, 如果一级 DTB 有条目淘汰, 则同样需将淘汰的条目写回二级 DTB。

需要注意的是: 当装填的 DTB 条目的地址和粒度与两级 DTB 中已有条目的地址和粒度有重叠时, 硬件不再保证将原 DTB 中的条目删除, 因此软件必须保证: 当需要更改某个页面的粒度或虚实地址对应关系时, 必须对 DTB 进行软件刷新, 否则硬件在进行 DTB 访问时, 可能因命中多个条目而出错。

对 DTB 条目的刷新方式有五种, 可通过写以下 CSR 来实现的:

- 1) 对指定虚页面的 DTB 条目刷新: 写 CSR: DTB_IS 实现;
- 2) 对指定用户进程号的 DTB 条目刷新: 写 CSR: DTB_IU 实现;
- 3) 对指定虚拟机号的全部用户进程的 DTB 条目刷新: 写 CSR: DTB_IVP 实现;
- 4) 对指定虚拟机号的 DTB 条目刷新: 写 CSR: DTB_IV 实现;
- 5) 对所有 DTB 条目刷新: 写 CSR: DTB_IA 实现。

执行这些刷新指令时, 硬件保证对两级 DTB 都进行刷新。

4.4 存储空间访问方式

4.4.1 虚空间

申威 411 处理器支持的虚拟地址为 43 位 (VA[42:0]), 并引入了 2 位虚拟机号 (VPN) 和 8 位用户进程号 (UPN), 将 43 位的虚地址空间延伸到 53 位, 其中 VPN 用于区分 4 个不同的虚拟机, UPN 用于区分同一虚拟机下的 256 个不同的进程。对 43 位的虚地址空间也进行了划分, 不同处理器模式下有不同的访问权限:

- 1) 用户模式 (UM): 只能访问 VA[42]为“0”的空间, 且必须采用虚地址访问, 不支持超页方式访问;
- 2) 内核模式 (KM): 只能访问 VA[42]为“1”的空间, 且必须采用虚地址访问, 不支持超页方式访问;
- 3) 虚拟模式 (VM): 可访问 VA[42]为“0”空间, 但只能采用超页方式访问; 也可访问 VA[42]

为“1”空间，但只能采用虚地址访问；

4) 硬件模式 (HM): 可以访问所有虚空间, 既可以使用虚地址访问, 也可以直接使用物理地址访问。

注意: 虚拟模式与用户模式的虚地址空间是重叠的, 由操作系统进行管理, 保证使用的虚地址不重叠。

每个虚拟机下的进程可以是用户进程 (包含一些系统管理进程), 也可以是内核 (OS) 进程, 不同类型的进程应运行在不同模式下。在一个核心所运行的环境中, 每个虚拟机同时只支持 256 个活动进程, 但用户和系统进程数可以远远大于 256, 256 个活动进程以外的其它进程处于“挂起”状态。处于“挂起”状态的进程被“激活”时, 必须从 256 个活动进程号中选择一个当前不用或者可以“挂起”的进程进行切换。实际进程号与核心内部的进程号不要求一一对应 (除非不超过 256 个), 在某个“活动”进程号释放给其它“挂起”的进程使用时, 必须刷新指令 Cache、ITB 和 DTB 中与该进程相关的所有痕迹。同样, 当 4 个虚拟机号不够用时, 也可以类似处理。在 256 个活动的进程中进行切换时, 则不需要刷新指令 Cache、ITB 和 DTB; 但在 4 个活动的虚拟机号中进行切换时, 则需要刷新指令 Cache, 因为指令 Cache 是虚地址访问, 且没有存放 VPN。

核心在硬件上是单进程控制的, 通过 CSR 指示当前运行的虚拟机号 VPN 和进程号 UPN。为了使用方便, 核心内部有 2 组 CSR 分别都含有 VPN 和 UPN, CSR: UPCR[UPN]和 CSR: VPCR[VPN]用于 ICache 访问以及 ITB 的查询、装填和刷新; CSR: DTB_PCR[VPN、UPN]用于 DTB 的查询、装填和刷新。一般情况下, 这两组必须完全相同。如果操作系统需要改变虚地址与物理地址的映射关系, 在改变对应关系之前, 需要刷新指令 Cache、ITB 和 DTB 中的相关内容。

操作系统 (内核) 程序是运行在内核模式下的系统管理程序 (一种特殊的进程), 为便于操作系统的存储空间管理, 提高处理效率, 设置了专门的 KS 标志。KS 标志为“1”的空间指示为内核程序访问空间, KS 标志为“0”的空间指示为用户程序访问空间。在 TLB 装填时, 使用 KS 标志区分用户进程和内核进程, KS 为“1”时, 指示内核进程, 否则由 UPN 来指示不同的用户进程。

4.4.2 访问方式

4.4.2.1 物理地址访问

申威 411 处理器支持直接使用物理地址进行指令流和数据流访问, 不进行虚实地址转换, 也不进行存储保护以及访问权限检查。物理地址访问包括以下几种情况:

1) HM 模式下的指令流访问: 当申威 411 处理器处于 HM 模式下, 所有指令流访问的地址都当作物理地址处理, 直接将 PC[38:2]位作为物理地址来进行指令流访问, 物理地址[39]位固定为“0”;

2) HM 模式下的执行 PRI_LDx/p、PRI_LDx_INC/p、PRI_LDx_DEC/p、PRI_LDx_SET/p 和 PRI_STx/p 指令时, 数据流访问的有效地址当作物理地址处理, 不进行地址对界检查, 直接将 VA[39:0]

位作为物理地址来进行访问；

3) VM 模式下的指令流超页访问：当申威 411 处理器处于 VM 模式下，指令流访问的虚地址[42]位为“0”时，直接将虚地址[38:2]位当作物理地址使用，物理地址[39]位固定为“0”；

4) VM 模式下的数据流超页访问：当申威 411 处理器处于 VM 模式下，数据流访问的虚地址[42]位为“0”时，数据流访问的有效地址当作物理地址处理，直接将虚地址[39:0]位当作物理地址使用。

4.4.2.2 虚地址访问

虚地址访问包括以下情况：

1) HM 模式下，执行 PRI_LDx/p、PRI_LDx_INC/p、PRI_LDx_DEC/p、PRI_LDx_SET/p 和 PRI_STx/p 以外的其它访存指令；

2) VM 模式下，指令流和数据流访问的虚地址[42]位为“1”；

3) KM 或 UM 模式下，任何指令流和数据流访问。

虚地址访问时，需要使用系统空间标志 KS、虚拟机号 VPN、用户进程号 UPN 与虚地址一起决定所访问的存储空间。KS 位用来区分系统空间和用户空间，当 KS 为“0”时，指示对应的空间为用户程序或数据，否则为系统程序或数据。VPN 用于区分不同的虚拟机，UPN 用来区别同一虚拟机上的不同进程。运行在同一虚拟机上的操作系统允许多个虚地址对应同一个物理地址，但某一时刻，如果 VPN 相同，且 KS=1 或者 UPN 相同，则对应的物理地址应是唯一的。

在申威 411 处理器中，CSR: UPCR[UPN]和 CSR: VPCR[VPN]用于指令 Cache 访问以及 ITB 的查询、装填和刷新；CSR: DTB_PCR[VPN、UPN]用于 DTB 的查询、装填和刷新。

由于指令 Cache 中没有存放 VPN，如果切换虚拟机号，则需要刷新指令 Cache。如果操作系统需要改变虚地址与物理地址的映射关系，在改变对应关系之前，需要刷新指令 Cache、ITB 和 DTB 中的相关内容。另外，修改这些状态位时，还要注意满足读写对应 CSR 的顺序限制，详见附录 A。

4.4.3 地址检查

对核心产生的指令流访问，只进行地址合法性检查。对数据流访问，需要进行地址合法性检查和地址对界检查。具体的检查方法如下：

1) 如果指令流地址匹配使能，即 CSR: IA_MATCH[EN]为“1”，则指令经过指令流水线的译码站台时，检查指令流地址、处理器模式、虚拟进程号和用户进程号，若与 CSR: IA_MATCH 内容匹配（同时受 CSR: IA_MSK 控制），则产生指令流故障；

2) 执行部件处理跳转指令时，检查转移指令的目标地址，若寄存器 Rb 中指示的跳转目标指令地址[63:43]位不是[42]位的符号扩展，表示符号扩展错，产生指令流故障；

3) 如果指令流目标地址匹配使能，即 CSR: IDA_MATCH[EN]为“1”，则在执行部件处理整数转移类指令时，检查转移指令的目标地址和目标处理器模式，若与 CSR: IDA_MATCH 内容匹配（同

时受 CSR: IDA_MSK 控制), 则产生指令流故障;

4) 执行部件处理访存指令时, 检查计算出的访存虚地址, 如果虚地址[63:43]位不是[42]位的符号扩展, 表示符号扩展错, 则产生数据流故障;

5) 数据 Cache 管理部件对访存指令的数据流地址与 CSR: DA_MATCH 内容进行匹配检查(同时受 CSR: DA_MSK 控制), 若 DA_MATCH[MATCH]为“1”且匹配, 或者 DA_MATCH[MATCH]为“0”且不匹配则产生数据流故障。地址比较时, 如果 CSR: DA_MATCH[PA]为“1”, 则只比较物理地址[39:0]位; 否则比较虚地址[42:0]位。访存指令选择地址匹配比较的方式如下:

- a. 如果 CSR: DA_MATCH[EN(1:0)]为“00₂”, 则不进行匹配检查;
- b. 如果 CSR: DA_MATCH[EN(1:0)]为“01₂”, 则对读指令产生的地址进行比较;
- c. 如果 CSR: DA_MATCH[EN(1:0)]为“10₂”, 则对写指令产生的地址进行比较;
- d. 如果 CSR: DA_MATCH[EN(1:0)]为“11₂”, 则对所有访存指令产生的地址进行比较。

6) 整数部件处理访存指令时, 对计算出的访存地址进行对界检查, 即根据指令确定的数据访问粒度来检查访问地址, 若地址不对界, 在产生不对界故障。不对界的情况如下:

- a. LDHU、STH 指令且数据流地址[0]位不为“0”;
- b. LDW、STW、FLDS、FSTS、LDSE、LDWE、LDW_INC、LDW_DEC、LDW_SET 指令且数据流地址[1:0]位不为全“0”;
- c. LDL、LDL_U、STL、FLDD、FSTD、LDDE、LDL_INC、LDL_DEC、LDL_SET 指令且数据流地址[2:0]位不为全“0”;
- d. VLDS、VSTS 指令且数据流地址[3:0]位不为全“0”;
- e. VLDD、VSTD 指令且数据流地址[4:0]位不为全“0”;
- f. VLDW_Ux、VLDS_Ux、VSTW_Ux、VSTS_Ux 指令且数据流地址[1:0]位不为全“0”;
- g. VLDD_Ux、VSTD_Ux 指令且数据流地址[2:0]位不为全“0”。

Cache 刷新指令和数据装填指令都不进行地址对界检查, 总是以一个 Cache 行为处理单位, 即将其数据流地址的最低 7 位[6:0]强制为全“0”。对于特权指令 PRI_LD 和 PRI_ST, 不管采用物理地址还是虚地址进行访问, 均不进行地址对界检查。

7) 数据管理部件根据指令的类型和数据访问粒度对代换出的物理地址进行合法性检查, 若地址非法, 则产生数据流故障。地址非法的情况如下:

- a. SIMD 扩展访存指令、原子操作指令、不可 Cache 访存指令的访存地址 PA[39]为“1”(表示访问 IO 空间)时;
- b. 访存指令的访存地址 PA[39]为“1”(表示访问 IO 空间), 且粒度是 128 位或 256 位时。

8) 存控、系统接口各设备处理数据流或指令流访问时, 对地址进行合法性检查, 读类地址非法, 产生数据流故障, 写类地址非法, 产生机器检查错中断。地址非法的情况如下:

- a. 访存地址超出芯片支持的主存容量时;
- b. 访问不存在的 IO 寄存器时;

c. 对不可写的 IO 寄存器执行写操作时（写存控部分的不可写的 IO 寄存器时，每有进行该项检查）。

4.4.4 指令 Cache 访问

指令流访问指令 Cache 时，根据 PC 的[12:7]位访问指令 Cache 的标记存储器（ITAG），然后根据对应单元记录的 PM、KS 和 UPN 来判断是否命中指令 Cache。具体方法如下：

1) 如果申威 411 处理器运行在 HM 模式下，指令流地址为物理地址，将 PC[38:13]位与指令 Cache 中的四路标记地址[38:13]位比较，若存在一路地址标记相同、PM 位为“00₂”且有效位有效，则表示命中，否则表示不命中；

2) 如果申威 411 处理器运行在 VM 模式下，将 PC[42:13]位与指令 Cache 中的四路标记地址[42:13]位比较，若存在一路地址标记相同、PM 位为“01₂”且有效位有效，则表示命中，否则表示不命中；

3) 如果申威 411 处理器运行在 KM 或 UM 模式下，将 PC[42:13]位与指令 Cache 中的四路标记地址[42:13]位比较，若存在一路地址标记相同，KS 为“1”或 UPN 与 CSR: UPCR[UPN]相同，PM 与当前处理器模式相同，且有效位有效，则表示命中，否则表示不命中。

注意：由于指令 Cache 中不包括虚拟机号 VPN，当更改虚拟机号时，必须刷新指令 Cache。

4.4.5 ITB 访问

ITB 中包含 KS、VPN 和 UPN 位，在 ITB 装填时写入，其中 VPN 来自 CSR: VPCR[VPN]，UPN 来自 CSR: UPCR[UPN]。当指令流不命中指令 Cache 时，查询 ITB，具体方法如下：

1) 若是 HM 模式下的指令流访问，或者 VM 模式下指令流访问且虚地址[42]位为“0”，则不需要进行虚地址代换，直接将虚地址[38:0]位作为物理地址（物理地址[39]位固定为“0”），并结束。否则到 2)；

2) 将指令流虚地址 IVA 与 ITB 中所有有效条目的地址标记进行比较，比较的位数根据 ITB 条目中的粒度 GH[1:0]来确定，GH 为 0、1、2 或 3，比较的虚地址分别为 IVA[42:13]、[42:18]、[42:23]或[42:28]位，如果没有匹配的 ITB 条目，则产生 ITB 脱靶，转 7)。否则到 3)；

3) 如果条目中的 VPN 与 CSR: VPCR[VPN]不同，则产生 ITB 脱靶，转 7)。否则转 4)；

4) 如果条目中的 VPN 与 CSR: VPCR[VPN]相同，KS 位为“0”，且 UPN 与 CSR: UPCR[UPN]不同，则产生 ITB 脱靶，转 7)。否则到 5)；

5) 进行访问权限检查，如果满足以下条件之一，则产生指令流故障，并结束。否则到 6)；

- a. PC[1:0]为“01₂”且命中的 ITB 条目中 VEE 为“0”；
- b. PC[1:0]为“10₂”且命中的 ITB 条目中 KEE 为“0”；
- c. PC[1:0]为“11₂”且命中的 ITB 条目中 UEE 为“0”。

6) ITB 命中，根据命中条目的粒度 GH[1:0]来获得物理地址，并结束。具体方法如下：

- a. 若 GH 为 “00₂”，则 IPA[38:0]由 ITB 中 PA[38:13]和 IVA[12:0]组成；
- b. 若 GH 为 “01₂”，则 IPA[38:0]由 ITB 中 PA[38:18]和 IVA[17:0]组成；
- c. 若 GH 为 “10₂”，则 IPA[38:0]由 ITB 中 PA[38:23]和 IVA[22:0]组成；
- d. 若 GH 为 “11₂”，则 IPA[38:0]由 ITB 中 PA[38:28]和 IVA[27:0]组成。

7) ITB 脱靶，根据指令流访问虚地址[42]位进行判断，如果为 “0”，则产生 ITB 用户脱靶 (ITB_MISS_0) 故障；如果为 “1”，则产生 ITB 核心脱靶 (ITB_MISS_1) 故障。

4.4.6 DTB 访问

DTB 中包含了 KS、VPN 和 UPN 位，在 DTB 装填时写入，其中 VPN 来自 CSR:DTB_PCR[VPN]，UPN 来自 CSR:DTB_PCR[UPN]。数据流先访问 DTB，得到物理地址后再访问数据 Cache，具体方法如下：

1) 若满足以下条件之一，则不需要进行虚实地址代换，直接将数据流虚地址 DVA[39:0]作为物理地址。否则到 2)；

a. HM 模式下，数据流访问地址由 PRI_LDx/p、PRI_LDx_INC/p、PRI_LDx_DEC/p、PRI_LDx_SET/p 和 PRI_STx/p 指令产生；

b. VM 模式下，数据流访问虚地址[42]位为 “0”。

2) 数据流访问虚地址 DVA 与 DTB 中所有有效条目的地址标记进行比较，比较的位数根据 DTB 条目中的粒度 GH[1:0]来确定，GH 为 0、1、2 或 3，比较的虚地址分别为 DVA[42:13]、[42:18]、[42:23] 或 [42:28]位，如果没有匹配的 DTB 条目，则产生 DTB 脱靶，转到 7)。否则到 3)；

3) 如果条目中的 VPN 与 CSR:DTB_PCR[VPN]不同，则产生 DTB 脱靶，转到 7)。否则转 4)；

4) 如果条目中的 VPN 与 CSR:DTB_PCR[VPN]相同，KS 位为 “0”，且 UPN 与 CSR:DTB_PCR[UPN]不同，则产生 DTB 脱靶，转到 7)。否则到 5)；

5) 根据数据流访问指令的 PC[1:0]位，进行访问权限检查，如果满足以下条件之一，则产生数据流故障，并结束。否则到 6)；

a. 对一般读访问类指令的读越权检查，读越权的条件是 PC[1:0]为 “00₂” 且命中的 DTB 条目中 HRE 为 “0”，或者 PC[1:0]为 “01₂” 且命中的 DTB 条目中 VRE 为 “0”，或者 PC[1:0]为 “10₂” 且命中的 DTB 条目中 KRE 为 “0”，或者 PC[1:0]为 “11₂” 且命中的 DTB 条目中 URE 为 “0”；

b. 对一般写访问类指令的写越权检查，写越权的条件是 PC[1:0]为 “00₂” 且命中的 DTB 条目中 HWE 为 “0”，或者 PC[1:0]为 “01₂” 且命中的 DTB 条目中 VWE 为 “0”，或者 PC[1:0]为 “10₂” 且命中的 DTB 条目中 KWE 为 “0”，或者 PC[1:0]为 “11₂” 且命中的 DTB 条目中 UWE 为 “0”；

c. 对 LDx_INC、LDx_DEC 和 LDx_SET 等非特权原子操作指令，既要进行读越权检查，也要进行写越权检查；

- d. 对 PRI_LDx/v 或 PRI_LDx/vpte 指令，类似于一般读访问指令进行读越权检查，并使用 CSR: DTB_PCR[PM]代替 PC[1:0]进行权限检查；
 - e. 对 PRI_STx/v，类似于一般写访问指令进行写越权检查，并使用 CSR: DTB_PCR[PM]代替 PC[1:0]进行权限检查；
 - f. 一般读访问指令、非特权原子操作指令和 PRI_LDx/v 或 PRI_LDx/vpte，命中的 DTB 条目中 FOR 为“1”，作为读故障；
 - g. 一般写访问指令、非特权原子操作指令和 PRI_STx/v 指令，命中的 DTB 条目中 FOW 为“1”，作为写故障；
- 6) DTB 命中，根据命中条目的粒度 GH[1:0]来获得物理地址，并结束。具体方法如下：
- a. 若 GH 为“00₂”，则 DPA[38:0]由 DTB 中 PA[38:13]和 DVA[12:0]组成；
 - b. 若 GH 为“01₂”，则 DPA[38:0]由 DTB 中 PA[38:18]和 DVA[17:0]组成；
 - c. 若 GH 为“10₂”，则 DPA[38:0]由 DTB 中 PA[38:23]和 DVA[22:0]组成；
 - d. 若 GH 为“11₂”，则 DPA[38:0]由 DTB 中 PA[38:28]和 DVA[27:0]组成。
- 7) DTB 脱靶：若是 PRI_LDx/vpte 指令产生的数据流虚地址，则作为 DTB 二次脱靶故障 (DTBM_DOUBLE)，否则进入 DTB 一次脱靶故障，并根据数据流虚地址[42]位进行判断，如果为“0”，则作为 DTB 用户脱靶故障 (DTBM_SINGLE_0)；如果为“1”，则作为 DTB 核心脱靶故障 (DTBM_SINGLE_1)。

4.4.7 存储器栏栅 (MEMB)

指令流水线发现存储器栏栅指令时，需要等前面所有访存指令都完成处理并退出后，才允许后续指令进入指令流水线，因此软件在需要严格保证两条访存指令之间的顺序，或者一条访存指令与后续其它指令的顺序时，可以在两条指令之间插入一条存储器栏栅指令。

4.4.8 指令栏栅 (IMEMB)

指令流水线发现指令栏栅指令时，需要等前面所有的指令都完成处理并退出后，才开始重新取后续指令进入指令流水线，因此软件在需要严格进行指令的生产者与消费者同步时，可结合使用存储器栏栅指令和指令栏栅指令来实现。

4.5 错误检测及错误处理

4.5.1 检错与报错

4.5.1.1 校验方法

在申威 411 处理器中采用 2 种校验方法，一种是偶校验，另一种是 ECC 校验。偶校验是使被校验的数据和校验位中“1”的个数保持为偶数，否则发生偶校验错。偶校验只能发现被校验的数据和校验位中出现奇数个错误，但不能进行错误纠正。ECC 校验可实现纠正单位错，检测双位错和部分多位错。两种校验位的形成方法如下：

1) 偶校验：P 为校验位， $D[i:0]$ 为数据，则 $P = D_0 \oplus D_1 \oplus \dots \oplus D_i$ ，其中“ \oplus ”为逻辑“异或”，即将 $i+1$ 位数据进行逻辑异或而获得校验位 P；

2) ECC 校验：P[n:0] 为校验位， $D[m:0]$ 为数据，根据实际需要，确定 m 的大小，从而确定 n 的大小，保证能够纠正“ $m+n+2$ ”位数据（包含 ECC 校验位在内）中存在的单位数据错，检测其中的双位和部分多位数据错。

出现校验错，根据不同的错误现场与引起错误的指令是否已完成处理并退出等情况，可产生数据流故障、已纠正错中断请求或机器检查错中断请求。

申威 411 处理器内部的 Cache 存储器以及主存都采用校验保护，具体如下：

1) 指令 Cache 的地址标记存储器 (ITAG)：采用偶校验，每个 Cache 行的每一组 ITAG 内容 (41 位) 增加 1 位偶校验位；

2) 指令 Cache 的数据存储器：采用偶校验，每条指令 (32 位指令及 3 位预译码位) 增加 1 位偶校验位；

3) 数据 Cache 的地址标记存储器 (DTAG)：采用偶校验，每个 Cache 行的每一组 DTAG 内容 (24 位) 增加 1 位偶校验位；

4) 数据 Cache 的数据存储器：采用 ECC 校验，每个对界的 64 位数据增加 8 位 ECC 校验位；

5) 二级 Cache 的地址标记存储器 (STAG)：采用 ECC 校验，每个 Cache 行的每一组 STAG 内容 (24 位) 增加 6 位 ECC 校验位；

6) 二级 Cache 的数据存储器：采用 ECC 校验，每个对界的 64 位数据增加 8 位 ECC 校验位；

7) 二级 Cache 的标记副本存储器 (CTAG)：采用 ECC 校验，每两组 CTAG 内容 (共 42 位) 增加 7 位 ECC 校验位；

8) 主存：采用 ECC 校验，每个对界的 64 位数据增加 8 位 ECC 校验位。

申威 411 处理器的主要数据通路也设置了校验，请求和数据通路都采用 ECC 校验，每个对界的 64 位数据增加 8 位 ECC 校验位，具体如下：

1) 核心与一致性处理部件之间的接口：核心接收请求时，只检错和报错（包括单错和多错），

不纠单错；核心接收数据时纠单错、检多错并进行单错预警和报多错；一致性处理部件接收请求时，只检错和报错（包括单错和多错），不纠单错；一致性处理部件接收数据时，只检错和报错（只包括多错），不纠单错；

2) 一致性处理部件与交叉开关之间的接口：一致性处理部件接收请求时，只检错和报错（包括单错和多错），不纠单错；一致性处理部件接收数据时，只检错和报错（只包括多错），不纠单错；交叉开关接收请求和数据时，不进行检错和报错；

3) 一致性处理部件与存控之间的接口：一致性处理部件接收数据时，只检错和报错（只包括多错），不纠单错；存控接收请求和数据时，纠单错、检多错并进行单错预警和报多错；

4) 存控与 DDR3 存储器之间的接口：存控接收数据时，实现纠单错、检多错并进行单错预警和报多错，但多错不重新生成校验码；

5) 交叉开关与协议转换部件之间的接口：协议转换部件接收请求时，纠单错、检多错并进行单错预警和报多错；协议转换部件接收到本地的数据时，纠单错、检多错并进行单错预警和报多错；协议转换部件接收到远程的数据时，不检也不纠；交叉开关接收请求和数据时，不进行检错和报错；

6) 协议转换部件与 PCI-E 和维护之间的接口：协议转换部件接收请求时，纠单错、检多错并进行单错预警和报多错；协议转换部件接收到本地的数据时，纠单错、检多错并进行单错预警和报多错；协议转换部件接收到远程的数据时，不检也不纠；PCI-E 和维护接收请求和数据时，实现纠单错、检多错并进行单错预警和报多错；

除主要的通路以外，申威 411 处理器还在维护串行通路上设置了校验，具体如下：

1) 维护与两个 PCI-E 之间的接口：请求和数据通路都采用串行接口传输，每传输 64 位信息，则增加 8 位 ECC 校验位，接收方实现纠单错、检多错并进行单错预警和报多错。

2) 维护接口与维护系统之间的接口：请求和数据通路都采用串行接口传输，每传输 8 位信息，则增加 1 位偶校验位，维护接口接收请求和数据时，进行检错和报错。

4.5.1.2 可纠正错预警机制

核心内设置了一套 CSR: HSERR_CNT 和 HSERR_TH，用于对核心内检查到的可纠正错统一进行加“1”计数。当在预设定的时间窗口内，核心内的可纠正单错发生的次数达到预设计的阈值，且已纠正错中断使能时，产生已纠正错中断，同时通知核外的维护接口部件。参与核心可纠正错预警计数的有如下情况：

- 1) 读 ICache，检查到偶校验错；
- 2) 读 ITag，检查到偶校验错；
- 3) 读 DTag，检查到偶校验错；
- 4) DCache 在执行读修改写时，检查到 ECC 单错；
- 5) 数据从 DCache 淘汰到 SCache 时，检查到 ECC 单错；

- 6) 读 SCache, 检查到 ECC 单错;
- 7) 读 STag, 检查到 ECC 单错;
- 8) 读 ICTag, 检查到偶校验错;
- 9) SCache 管理部件 (SBOX) 收到核心接口上的响应数据, 检查到 ECC 单错。

4.5.1.3 核心的报错类型

核心通过维护接口向外报告的错误类型如下:

- 1) 核心内检查到的硬件故障, 包括:
 - A、IBOX 检查到 ROB 的头尾指针管理错误;
 - B、DBOX 检查到 CSR 指令发射错误;
 - C、SBOX 检查到 STAG 为非法状态;
 - D、SBOX 检查到核心接口上收到的二次请求与内部状态不一致;
 - E、SBOX 检查到核心接口上收到的响应与内部状态不一致;
 - F、SBOX 检查到核心接口上的响应包头和二次请求包头发生了 ECC 单错或多错;
 - G、软件自己检查到各种硬件故障, 设置 CSR: HARD_ERR 为“1”。
- 2) 核心内检查到的软件故障, 包括:
 - A、软件自己检查到各种软件故障, 设置 CSR: SOFT_ERR 为“1”;
 - B、写不可 Cache 空间或 I/O 空间时, 核心收到非法地址响应。
- 3) 核心内检查到的可纠错计数到达阈值, 包括 4.5.1.2 中列举的所有情况;
- 4) 核心内检查到的 ECC 多错, 包括:
 - A、DCache 在执行读修改写时, 检查到不可覆盖的 ECC 多错;
 - B、数据从 DCache 淘汰到 SCache 时, 检查到 ECC 多错;
 - C、读 SCache, 检查到 ECC 多错;
 - D、读 STag, 检查到 ECC 多错;
 - E、SBOX 从核心接口上收到响应数据, 检查到 ECC 多错。

4.5.2 指令 Cache 的错误检测与处理

指令 Cache 具有只读特性, 其数据或 ITAG 产生的偶校验错都是可纠正错。取指令时, 如果发现指令 Cache 的数据或 ITAG 出现偶校验错, 硬件分别设置 CSR: IS_STAT[DATA_PAR]或 CSR: IS_STAT[TAG_PAR], 并自动将指令 Cache 和指令副本标记阵列中, 四路对应的 Cache 行刷新, 同时参与可纠正错预警计数。

4.5.3 数据 Cache 的错误检测与处理

数据 Cache 的 DTAG 采用偶校验,当处理 Load/Store 类指令时,对从数据 Cache 中读出的 DTAG 进行偶校验,若产生偶校验错,则在 CSR: DC_CTL[DCTAG_PAR_EN]为“1”时,设置 CSR: DS_STAT[TAG_PERR]为“1”。该错误硬件无法自动纠正,需要进入数据流故障自陷(DFAULT),由 DFAULT 特权程序来进行错误恢复。

特权程序先读 CSR: DS_STAT,确定引起 DFAULT 的原因,如果不存在其它数据流故障且 DS_STAT[TAG_PERR]为“1”,则可以确定是数据 Cache 的 DTAG 偶校验错,然后设置 CSR: DC_CTL[DCTAG_PAR_EN]为“0”,关闭数据 Cache 的 DTAG 校验使能,再从 CSR: DVA 中获得出现错误的物理地址。根据这个物理地址[14:7]位,形成 32 个索引地址相同,二级 Cache 组号不同的淘汰地址,使用 FLUSHD 指令将出现故障的数据 Cache 内容淘汰到主存。由于出现 DTAG 偶校验错的数据 Cache 行被淘汰时,由 CTAG 和 STAG 提供地址标记信息,因此数据 Cache 的 DTAG 偶校验错被清除,且对应的数据 Cache 内容不会受到破坏。

数据 Cache 的数据采用 ECC 校验,当处理 Load 类指令时,对从数据 Cache 中读出的数据进行 ECC 校验,如果出现 ECC 错(硬件不判断是单错还是多错),则在 CSR: DC_CTL[DCDAT_ERR_EN]为“1”时,设置 CSR: DS_STAT[DAT_ERR]为“1”。该错误硬件无法自动纠正,需要进入数据流故障自陷(DFAULT),由 DFAULT 特权程序来进行处理。

特权程序先读 CSR: DS_STAT,确定引起 DFAULT 的原因,如果不存在其它数据流故障且 DS_STAT[DAT_ERR]为“1”,则可以确定是数据 Cache 的数据 ECC 校验错,然后设置 CSR: DC_CTL[DCDAT_PAR_EN]为“0”,关闭数据 Cache 的数据校验使能,再从 CSR: DVA 中获得出现错误的物理地址,使用 8 条普通的 load 指令,其访问地址的[12:7]与 DVA[12:7]相同,将出现故障的数据 Cache 内容通过挤占方式淘汰到 SCache。出现 ECC 校验错的数据 Cache 行被淘汰时,如果是脏数据,二级 Cache 管理部件会对数据进行 ECC 校验,纠正单错,报告多错,否则数据 Cache 会丢弃非脏的出错数据,实现纠正单错和报告多错。

数据 Cache 处理 Store 类指令时,对从数据 Cache 中读出的数据进行 ECC 校验,如果出现 ECC 单错,则进行纠错,并报告 ECC 单错;如果出现能被写数据完全覆盖的 ECC 多错,则直接用新数据覆盖,不报错;如果出现不能被写数据完全覆盖的 ECC 多错,则硬件无法纠正,在 CSR: DC_CTL[DCDATA_ERR_EN]为“1”时,设置 CSR: DC_STAT[DATA_MERR]为“1”,并报告 ECC 多错。

二级 Cache 对从数据 Cache 中淘汰的数据进行 ECC 校验,如果出现 ECC 单错,则进行纠错,在 SC_CTL[SCDAT_ERR_EN]为“1”时,设置 CSR: SC_STAT[DVIC_SERR]为“1”,并报告 ECC 单错;如果出现 ECC 多错,则硬件无法纠正,在 CSR: SC_CTL[SCDAT_ERR_EN]为“1”时,设置 CSR: SC_STAT[DVIC_MERR]为“1”,并报告 ECC 多错。

以上情况下,对于报出的 DTAG 偶校验错和 ECC 单错,同时参与可纠正错预警计数;对于报

出的 ECC 多错，则在机器检查错使能的情况下，产生机器检查错中断请求。

4.5.4 二级 Cache 的错误检测与处理

二级 Cache 处理指令流访问、数据流访问以及一致性查询请求时，对从二级 Cache 中读出的 STAG 进行 ECC 校验，如果出现 ECC 单错，则进行纠正，在 CSR: SC_CTL[SCTAG_ERR_EN]为“1”时，设置 CSR: SC_STAT[TAG_SERR]为“1”，并报告 ECC 单错；如果产生 ECC 多错，则硬件无法纠正，在 CSR: SC_CTL[SCTAG_ERR_EN]为“1”时，设置 CSR: SC_STAT[TAG_MERR]为“1”，并报告 ECC 多错。

二级 Cache 处理指令流访问和一致性查询请求时，对从二级 Cache 中读出的数据进行 ECC 校验，如果出现 ECC 单错，则进行纠正，在 CSR: SC_CTL[SCDAT_ERR_EN]为“1”时，设置 CSR: SC_STAT[DAT_SERR]为“1”，并报告 ECC 单错；

二级 Cache 处理数据流请求时，对从二级 Cache 中读出的数据进行 ECC 校验，如果出现 ECC 单错，则进行纠正后写入 DCache，但可能存在需要写寄存器文件的 Load 指令，此时其数据未进行纠错，硬件会重新执行这条指令，从而从 DCache 读取正确的数据；同样的，二级 Cache 在 CSR: SC_CTL[SCDAT_ERR_EN]为“1”时，设置 CSR: SC_STAT[DAT_SERR]为“1”，并报告 ECC 单错；

如果出现 ECC 多错，则硬件无法纠正，在 CSR: SC_CTL[SCDAT_ERR_EN]为“1”时，设置 CSR: SC_STAT[DAT_MERR]为“1”，并报告 ECC 多错。

以上情况下，对于报出 ECC 单错，都参与可纠正错预警计数；对于报出 ECC 多错，则在机器检查错使能的情况下，产生机器检查错中断请求。

4.5.5 指令副本标记的错误检测与处理

指令 Cache 具有只读特性，指令副本标记阵列产生的偶校验错也是可纠正错。正常访问过程中，如果发现指令副本标记阵列出现偶校验错，则硬件设置 CSR: SC_STAT[ICTAG_PAR]，并自动将指令 Cache 和指令副本标记阵列中，四路对应的 Cache 行刷新，同时参与核心可纠正错预警计数。

4.5.6 核心接口上的错误检测与处理

核心的二级 Cache 管理部件 (SBOX) 对核心收到的二次请求进行偶校验，如果发生偶校验错，且 CSR: SC_CTL[SYSREQ_ERR_EN]为“1”，则登记 CSR: SC_STAT[SYSREQ_ERR]，且向 IBOX 报告错误报告，在机器检查错使能的情况下，产生机器检查错中断请求；如果 CSR: SC_CTL[SYSREQ_ERR_EN]为“0”，则不登记任何信息，也不向 IBOX 报告错误。

SBOX 对核心收到的响应包头信息进行偶校验，如果发生偶校验错，且 CSR: SC_CTL[SYSACK_ERR_EN]为“1”，则登记 CSR: SC_STAT[SYSACK_ERR]，且向 IBOX 报告错

误，在机器检查错使能的情况下，产生机器检查错中断请求；如果 CSR: SC_CTL[SYSACK_ERR_EN] 为“0”，则不登记任何信息，也不向 IBOX 报告错误。

SBOX 对核心收到的响应数据进行 ECC 校验，如果数据发生了单错，则进行纠错，如果 CSR: SC_CTL[SYSDAT_ERR_EN]为“1”，则登记 CSR: SC_STAT[SYSDAT_SERR]，并参与核心可纠正错预警计数；如果 CSR: SC_CTL[SYSDAT_ERR_EN]为“0”，则不登记任何信息，也不参与核心可纠正错预警计数。

SBOX 对核心收到的响应数据进行 ECC 校验，如果数据发生了多错，且 CSR: SC_CTL[SYSDAT_ERR_EN]为“1”，则登记 CSR: SC_STAT[SYSDAT_MERR]，然后将错误报告给 IBOX，在响应中携带 ECC 多错标志送 IBOX 或 DBOX，同时错误的重新生成校验码后装填 SCache；如果 CSR: SC_CTL[SYSDAT_ERR_EN]为“0”，则不登记任何信息，也不在响应中携带 ECC 多错标志。

如果 DBOX 收到带 ECC 多错标志的响应，且是非推测路径上的数据流访问请求，且产生访问请求的指令不是数据装填，则设置 CSR: DC_STAT[MEM_MERR]为“1”，进入数据流故障处理入口执行，同时重新生成 ECC 码装填 DCache。

如果 IBOX 收到带 ECC 多错标志的响应，且是非推测路径上的取指令请求，则设置 CSR: IS_STAT[MEM_MERR]为“1”，如果机器检查错使能，则作为机器检查错中断处理，否则继续取指，此时执行的是特殊的 SYS_CALL 指令（功能码为 0xBB），错误的重新生成校验码后装填 ICache。

4.5.7 核心响应中携带的错误

申威 411 处理器访问核外存储器空间或 I/O 空间，可能收到三种响应：

- 1) 非法地址响应；
- 2) 带控制错标志的响应。
- 3) 带 ECC 多错标志的响应。

4.5.7.1 非法地址响应

核心向存储控制器发出数据读或指令读请求，存储控制器因请求包含的地址是不存在地址空间的地址（NXM），而产生非法地址响应返回核心。或者核心向存储控制器或系统接口发出 I/O 读请求，所访问的 I/O 地址不存在，则这些部件也产生非法地址响应返回核心。如果请求是指令推测执行产生的，则不产生任何影响，否则作为软件故障，根据请求类型作出不同处理：

1) 如果请求是取指令请求，则设置 CSR: IS_STAT[ADDR_ERR]为“1”，如果机器检查错使能，则进入机器检查错中断处理入口，否则继续执行特殊的 SYS_CALL 指令（功能码为 0xBB），最终处于停机状态；2) 如果请求是数据流访问请求，则设置 CSR: DS_STAT[ACV1]为“1”，作为数据流故障处理。

注意，产生非法地址响应错时，将不装填数据 Cache（数据流访问）、指令 Cache（指令流访问）和二级 Cache，且将数据 Cache 和二级 Cache 对应的 Cache 行状态置为无效。软件进入机器检查错中断处理程序或数据流故障处理程序后，检查是这种情况引起的故障，需要设置 CSR: SOFT_ERR 为“1”，以便向核外报告本核心发生软件故障。

核心发出的 I/O 空间写请求、共享不可 Cache 空间写请求或者挤占 SCache 引起的淘汰请求，因请求包含的地址是不存在地址空间的地址（NXM），或所访问的 IOR 不可写，则产生非法地址响应返回核心。如果 CSR: SC_CTL[WR_ERR_EN]为“1”，则设置 CSR: SC_STAT[WR_ADDR_ERR]为“1”。如果机器检查错使能，则作为机器检查错中断处理，否则继续执行。同时硬件向核外报告本核心发生了软件故障。

注 1：每个存控对应的存储器空间范围与外接的存储器容量密切相关，一致性处理部件会根据具体的容量配置判断所访问的地址是否非法；

注 2：当核心或存控被配置为断连状态时，对应的存储器地址或 I/O 寄存器地址也可能变为不存在的空间，一致性处理器部件和各设备接口部件会根据断连配置判断所访问的地址是否非法。

4.5.7.2 带控制错的响应

核心向存储控制器发出请求，存储控制器返回带控制错的响应，且 CSR: SC_CTL[ACK_CHK_EN]为“1”，则作为非法响应，并根据请求源类型作出不同处理：

1) 如果请求是非推测路径上的取指令请求，则设置 CSR: IS_STAT[ACK_ERR]为“1”，如果机器检查错使能，则作为机器检查错中断处理，否则继续取指，此时执行的是特殊的 SYS_CALL 指令（功能码为 0xBB），最终处于停机状态；

2) 如果请求是非推测路径上的数据流访问请求，当 CSR: DC_CTL[ACK_ERR_EN]为“1”，且产生访问请求的指令不是数据装填指令，则设置 CSR: DC_STAT[ACK_ERR]为“1”。如果机器检查错使能，则进入机器检查错中断处理入口执行，否则正常执行。

3) 如果请求是 I/O 空间写请求、共享不可 Cache 空间写请求或者挤占 SCache 引起的淘汰请求，当 CSR: SC_CTL[WR_ERR_EN]为“1”，则设置 CSR: SC_STAT[WR_CTL_ERR]为“1”。如果机器检查错使能，则作为机器检查错中断处理，否则继续执行。

收到带控制错的响应时，如果是带数据的响应，则装填 DCache（数据流访问）和 SCache，但不装填 ICache（指令流访问）。如果 CSR: SC_CTL[ACK_CHK_EN]为“0”，则不进入任何异常，继续执行。

4.5.7.3 带 ECC 多错的响应

核心向存储控制器发出请求，存储控制器返回带 ECC 多错的响应，作为非法响应，并根据请求

源类型作出不同处理:

1) 如果请求是非推测路径上的取指令请求, 则将虚地址登记到 CSR: EXC_PC, 设置 CSR: IS_STAT[MEM_MERR]为“1”, 如果机器检查错使能, 则进入机器检查错中断处理入口, 否则继续执行特殊的 SYS_CALL 指令(功能码为 0xBB), 最终处于停机状态;

2) 如果请求是非推测路径上的数据流访问请求, 则将指令的虚地址登记到 CSR: EXC_PC 中, 将访问的物理地址登记到 CSR: DVA 中, 设置 CSR: DC_STAT[MEM_MERR]为“1”, 然后进入数据流故障处理流程。

4.5.8 核心内其它硬件故障的检测与处理

核心硬件出现故障可能产生多种控制错误, 都当作机器检查错, 如果机器检查错中断使能, 则产生机器检测错中断请求。

4.5.8.1 指令流控制错

核心进行指令流处理时, 若重排序缓冲(ROB)中的头尾指针管理出现混乱时, 则作为指令流控制错, 设置 CSR: IS_STAT[ROB_ERR]为“1”。

4.5.8.2 数据流控制错

核心的内部检测到以下几种情况时, 产生数据流控制错:

1) 数据 Cache 管理部件(DBOX)接收到的一致性查询请求与内部保留的状态不一致, 且 DC_CTL[PROBE_ERR_EN]为“1”, 设置 CSR: DC_STAT[PROBE_ERR]为“1”;

2) 数据 Cache 管理部件(DBOX)中保留的一条 CSR 写指令还未退出, 又接收到一条新的 CSR 写指令, 且 DC_CTL[CSR_ERR_EN]为“1”, 设置 CSR: DC_STAT[CSR_ERR]为“1”;

3) 二级 Cache 管理部件(SBOX)收到响应或二次请求, 检查到与内部状态不一致时, 且 CSR: SC_CTL[CTL_ERR_EN]为“1”, 设置 CSR: SC_STAT[CTL_ERR]为“1”;

4) 二级 Cache 管理部件(SBOX)查询 STag, 检查到非法状态, 且 CSR: SC_CTL[CACHE_ST_ERR_EN]为“1”, 设置 CSR: SC_STAT[CTL_ERR]为“1”。

5 中断

5.1 中断源

中断是指来自于正常指令流之外的、与指令流异步的、需要申威 411 处理器进行干预和处理的事件。在申威 411 处理器中，设置四类中断，具体如下：

1) 核间中断：由本核心或外部其它发出的中断，用于实现申威 411 处理器之间的快速通信，可屏蔽；

2) 核心内部产生的核心中断：

- a. 已纠正错中断：核心硬件发现已纠正错误而产生，可屏蔽；
- b. 机器检查错中断：核心硬件发现故障时产生，可屏蔽；
- c. 定时器中断：核内工作时钟计数产生，可屏蔽；
- d. 事件计数中断：核内进行事件计数产生，可屏蔽；

3) 核心外部产生的核心中断：

- a. 设备中断：由芯片连接的外设产生的中断，包括 PCI-E 设备中断和维护中断，可屏蔽；
- b. 短时钟中断：核外维护时钟计数产生，可屏蔽；
- c. 故障中断：核外硬件发现故障时产生，可屏蔽；

4) 特殊中断：

- a. 睡眠中断：一种特殊中断，强制核心进入睡眠状态，可屏蔽；
- b. 睡眠唤醒中断：一种特殊中断，使核心结束睡眠状态，恢复运行，不可屏蔽；
- c. 复位：一种特殊中断，使处理器处于复位状态，不可屏蔽。

在申威 411 处理器中，内部 CSR: II0 和 II1 中记录有本核心收到的核间中断请求，CSR: INT_STAT0~3 中记录核心内部和核心外部产生的中断请求，特殊中断请求不需要在内部 CSR 中记录。核心内部和外部产生的中断都采用向量方式管理，即各种中断在 CSR: INT_STAT0~3 中的位置（称为中断向量号）可以由软件通过写对应的 CSR 或 IOR 来指定，核心内部产生的中断映射的向量号在 CSR: INT_VEC 中定义，核心外部产生的中断请求映射的向量号由对应的 IOR 来定义。除复位为电平触发的中断外，其它中断都是边沿触发类型的中断。

注意 CSR: INT_STAT0~3 显式读的特殊性，不论对应的中断使能（CSR: IER0~3 控制）是否打开，产生核心中断请求总是在 CSR: INT_STAT0~3 中进行记录（除复位、睡眠唤醒中断和核间中断），但显式读 CSR: INT_STAT0~3，只能获得那些对应中断使能打开的核心中断请求信息。

5.1.1 核间中断

5.1.1.1 核间中断 0/1

申威 411 处理器可以在任何模式下对一个或任意多个核心发出核间中断。当软件写 CSR: INT_REQ 时, 如果指定中断类型为核间中断, 则向系统接口部件发送核间中断请求, 可以指定发送核间中断 0 请求或核间中断 1 请求。

系统接口部件接到核间中断请求后, 检查指定的目标核心状态, 如果可处理中断且中断使能 (IOR: INTEN 为“1”), 则向该核心发核间中断请求; 如果核心睡眠或中断不使能 (IOR: INTEN 为“0”), 则将核间中断请求缓存在系统接口部件, 等核心可以处理中断后再向该核心发中断请求; 如果核心断连, 则核间中断丢失, 在 IOR: IINT_MIS 中进行登记, 同时通知维护接口发生了软件错, 登记 IOR: SI_FAULT_STAT, 可配置产生故障中断和向系统报错。

目标核心接到核间中断 0 请求后, 根据源核心号置 CSR: II0 中对应的 Valid 为“1”, 如果核间中断 0 使能 (CSR: II_ER[II0_EN]为“1”), 则进入一般中断处理的特权程序入口 0 (INTERRUPT0) 进行处理。当核间中断服务程序处理结束后, 软件写中断目标核心的 CSR: II_CLR[II0_CLR], 清除核间中断 0。

同样目标核心接到核间中断 1 请求后, 根据源核心号置 CSR: III 中对应的 Valid 为“1”, 如果核间中断 1 使能 (CSR: II_ER[III_EN]为“1”), 则进入一般中断处理的特权程序入口 0 (INTERRUPT0) 进行处理。当核间中断服务程序处理结束后, 软件写中断目标核心的 CSR: II_CLR[III_CLR], 清除核间中断 1。

需要注意的是, 申威 411 处理器可能将同一个源核心发向同一个目标核心的多次核间中断合并。另外, 硬件保证, 当核间中断到达核心时, 如果对应的中断使能没有打开 (即 CSR: II_ER[II_EN]为“0”), 则 CSR: II0 和 III 中一致保留中断, 直到被软件清除或被新的核间中断请求覆盖为止。

5.1.1.2 核间异步消息中断

申威 411 处理器可以在任何模式下对某个核心发异步消息, 在系统接口的 IO 空间上, 以本核心或其它核心为目标, 定义了两个 IOR: MAIL_BOX_i 和 MAIL_STAT_i (i 为 0~15)。当核心 n 想往核心 m 发异步消息时, 可通过写 IOR: MAIL_BOX_m 发送一个 64 位的消息。

系统接口收到该消息后, 先检查指定的目标核心状态, 如果可处理中断且中断使能 (IOR: INTEN 为“1”), 系统接口自动向核心 m 发一个核间异步消息中断请求; 如果核心睡眠或中断不使能 (IOR: INTEN 为“0”), 则将核间异步消息中断请求缓存在系统接口部件, 等核心可以处理中断后再向该核心发中断请求; 如果核心断连, 则向源核心返回非法地址响应。前两种情况下, 如果 IOR: MAIL_STAT_m[MAIL_NUM]指示邮箱不满, 则将消息保存。如果 IOR: MAIL_STAT_m[MAIL_NUM]

指示邮箱满，则置 IOR: MAIL_STAT_m[FULL_FLAG[n]] 为 1，指示本次消息丢失。

核心 m 收到异步消息中断请求后，内部计数器 (CSR: II_MAIL[MAIL_NUM]) 进行加“1”计数，如果 CSR: II_MAIL[MAIL_NUM] 中的值大于“0”，且 CSR: II_ER[II_MAIL_EN] 为“1”，则申请进入一般中断处理的特权程序入口 0 (INTERRUPT0) 进行处理。在中断处理程序中，通过读 IOR: MAIL_BOX_m 获得消息信息；成功读走一个消息后，IOR: MAIL_STAT_m[MAIL_NUM] 自动减“1”，CSR: II_MAIL[MAIL_NUM] 也自动减“1”。当 IOR: MAIL_STAT_m[MAIL_NUM] 为“0”时，读不成功，返回一个全“0”的消息。

软件必须保证源核心每发一次消息，都要检查 IOR: MAIL_STAT_m[FULL_FLAG[n]] 是否为“1”，如果为“1”，则说明刚发出的消息不成功，需要重发。另外，核心只允许读本核心收到的异步消息，如果读其它核心收到的异步消息将产生错误响应。

5.1.2 核内产生的中断

5.1.2.1 已纠正错中断

申威 411 处理器对内部存储器中出现的可纠正错实现了一种带预警功能的已纠正错中断，核心设置 CSR: HSERR_CNT、CSR: HSERR_TH，记录对 ICACHE、DCACHE、SCACHE 进行相关操作时产生可纠正错的情况。

当核心中的存储器在指定时间里发生可纠正错的次数超过指定阈值时，置 CSR: HSERR_CNT[SERR_OF] 为“1”，如果 CSR: IER0~3 中对应的已纠正错中断使能，则硬件自动在 CSR: INT_STAT0~3 中登记，登记的位置由 CSR: INT_VEC[CR_VEC] 决定，并进入一般中断处理特权程序入口 1 (INTERRUPT1) 进行处理。当已纠正错中断服务程序处理结束后，软件可通过写 CSR: INT_CLR0~3 中对应位来清除已纠正错中断，同时通过写 CSR: HSERR_CNT，清除原来的溢出标志。核心产生已纠正错的原因有以下几种：

- 1) 指令 Cache 的标记存储器出现偶校验错，此时 CSR: IS_STAT[TAG_PAR] 为“1”；
- 2) 指令 Cache 的数据存储器出现偶校验错，此时 CSR: IS_STAT[DATA_PAR] 为“1”；
- 3) 数据 Cache 的标记存储器出现偶校验错，此时 CSR: DS_STAT[TAG_PERR] 为“1”；
- 4) 处理 Store 指令时，读出的数据 Cache 数据出现 ECC 单错，此时不设置任何 CSR；
- 5) 数据 Cache 被淘汰或置换的数据出现 ECC 单错，此时 CSR: SC_STAT[DVIC_SERR] 为“1”；
- 6) 二级 Cache 的标记存储器出现 ECC 单错，此时 CSR: SC_STAT[TAG_SERR] 为“1”；
- 7) 二级 Cache 的数据存储器出现 ECC 单错，此时 CSR: SC_STAT[DAT_SERR] 为“1”；
- 8) 指令副本标记存储器出现偶校验错，此时 CSR: SC_STAT[ICTAG_PAR] 为“1”。

5.1.2.2 机器检查错中断

机器检查错是申威 411 处理器硬件产生各种控制错的总称，不同的机器检查错对申威 411 处理器的影响程度也不相同，轻者可恢复系统的运行，重者不可恢复，会导致系统崩溃。机器检查错中断分为精确中断和非精确中断，对应于同一个异常入口。

当出现机器检查错时，硬件自动在相关的 CSR 中记录具体的错误标志，如果 CSR: IER0~3 中对应的机器检查错中断使能，则硬件自动在 CSR: INT_STAT0~3 中登记机器检查错标志，登记的位置由 CSR: INT_VEC[MCHK_VEC]决定，并产生机器检查错中断请求，进入机器检查错中断处理特权程序入口（MCHK）执行。当中断服务程序处理结束时，可通过写 CSR: INT_CLR0~3 中对应位，清除机器检查错中断。

产生机器检查错的原因有以下几种，可分为精确中断与非精确中断两种：

- 1) 指令部件中的指令重排序缓冲出现头尾指针管理错，此时 CSR: IS_STAT[ROB_ERR]为“1”；（非精确中断）
- 2) 指令流访问核外主存时，收到的响应中带控制错标志，此时 CSR: IS_STAT[ACK_ERR]为“1”；（精确中断）
- 3) 指令流访问核外主存时，收到非法地址响应，此时 CSR: IS_STAT[ADDR_ERR]为“1”；（精确中断）
- 4) 指令流访问核外主存时，收到的响应带 ECC 多错，此时 CSR: IS_STAT[MEM_MERR]为“1”；（精确中断）
- 5) 数据流读访问核外主存时，收到的响应中带控制错标志，此时 CSR: DC_STAT[ACK_ERR]为“1”；（精确中断）
- 6) 数据 Cache 管理部件处理 Store 指令时，从数据 Cache 中读出的数据出现不能被写数据完全覆盖的 ECC 多错，此时 CSR: DC_STAT[DAT_MERR]为“1”；（非精确中断）
- 7) 数据 Cache 管理部件（DBOX）接收到的一致性查询请求与内部保留的状态不一致，此时 CSR: DC_STAT[PROBE_ERR]为“1”；（非精确中断）
- 8) 数据 Cache 淘汰的数据产生 ECC 多错，此时 CSR: SC_STAT[DVIC_MERR]为“1”；（非精确中断）
- 9) 二级 Cache 的 STAG 产生 ECC 多错，此时 CSR: SC_STAT[TAG_MERR]为“1”；（非精确中断）
- 10) 二级 Cache 的数据产生 ECC 多错，此时 CSR: SC_STAT[DAT_MERR]为“1”；（非精确中断）
- 11) 二级 Cache 从核外收到的二次请求发生偶校验错，此时 CSR: SC_STAT[SYSREQ_ERR]为“1”；（非精确中断）
- 12) 二级 Cache 从核外收到的响应包头发生偶校验错，此时 CSR: SC_STAT[SYSACK_ERR]

为“1”；（非精确中断）

13) 二级 Cache 管理部件 (SBOX) 检查到从核外收到的响应或二次请求与内部寄存的状态不一致, 此时 CSR: SC_STAT[CTL_ERR]为“1”；（非精确中断）

14) 二级 Cache 管理部件 (SBOX) 查询 STAG 发现状态非法, 此时 SC_STAT[CACHE_ST_ERR]为“1”；（非精确中断）

15) 对 I/O 空间或者共享不可 Cache 执行写访问, 或者二级 Cache 淘汰请求收到非法地址响应, 此时 CSR: SC_STAT[WR_ADDR_ERR]为“1”；（非精确中断）

16) 对 I/O 空间或者共享不可 Cache 执行写访问, 或者二级 Cache 淘汰请求收到的响应带控制错标志, 此时 CSR: SC_STAT[WR_CTL_ERR]为“1”。（非精确中断）

5.1.2.3 定时器中断

核心内部设置一个 64 位的计数器, 当 CSR: TIMER_CTL[TIMER_EN]为“1”时, 计数器就从 0 开始, 每拍进行加“1”计数, 当计数器达到或超过 CSR: TIMER_TH 指定的初值时, 便产生定时器中断, 同时将 CSR: TIMER_CTL[TIMER_EN]设置为 0。

产生定时器中断时, 如果 CSR: IER0~3 中对应的定时器中断使能, 则硬件自动在 CSR: INT_STAT0~3 中登记, 登记的位置由 CSR: INT_VEC[TIMER_VEC]决定, 并进入一般中断处理特权程序入口 1 (INTERRUPT1)。当定时器中断服务程序处理结束后, 软件可通过写 CSR: INT_CLR0~3 中对应位来清除定时器中断。

当核心处于睡眠状态时, 定时器对应的 CSR 和计数器中的值都被清 0, 如果需要重新启动定时器, 则需要重新设置 TIMER_CTL[TIMER_EN]和 CSR: TIMER_TH。

5.1.2.4 事件计数中断

申威 411 处理器设置了两个事件计数中断, 用于实现事件计数, 具体参看第八章。当硬件对指定模式下的特殊事件进行事件计数并产生溢出时, 产生事件计数 0 中断或事件计数 1 中断, 如果 CSR: IER0~3 中对应的事件计数中断使能, 则硬件自动在 CSR: INT_STAT0~3 中登记, 登记的位置由 CSR: INT_VEC[PC0_VEC]或 CSR: INT_VEC[PC1_VEC]决定, 并进入一般中断处理特权程序入口 1 (INTERRUPT1)。当事件计数中断服务程序处理结束后, 软件可通过写 CSR: INT_CLR0~3 中对应位来清除事件计数 0 或事件计数 1 中断。

5.1.3 核外产生的中断

申威 411 处理器目前定义了五类核外产生的中断, 这些中断请求都是通过申威 411 处理器的系统接口部件汇集后发送到各目标核心。每个核心支持的中断向量数为 256 个, 因此操作系统需要将

这些外部中断分配到不同核心中的不同向量号上，以免产生中断向量号重叠。具体的中断类型如下：

- 1) 2 路 PCI-E 接口产生的 INTA、INTB、INTC、INTD 中断；
- 2) 2 路 PCI-E 接口产生的 MSI 中断，每套接口最多支持 256 个 MSI 中断；
- 3) 系统接口部件产生的短时钟中断；
- 4) 维护接口产生的维护中断；
- 5) 维护接口产生的故障中断。

当某个核心收到系统接口发来的中断请求时，如果 CSR: IER0~3 中对应的外部中断使能，则在 CSR: INT_STAT0~3 中登记，并进入一般中断处理特权程序入口 1 (INTERRUPT1) 进行处理。当外部中断服务程序处理结束后，软件写中断目标核心的 CSR: INT_CLR0~3 中对应位，清除 CSR: INT_STAT0~3 中记录的中断请求。

需要注意的是，申威 411 处理器可能将相同向量号的外部中断合并。另外，硬件保证，当外部中断到达核心时，如果对应的中断使能没有打开（即 CSR:II_ER[II_EN]为“0”），则硬件保留该中断请求，直到被软件写 CSR: INT_CLR0~3 清除为止。

5.1.3.1 PCI-E 的 INTx 中断

每路 PCI-E 接口的 INTx 中断包括：INTA、INTB、INTC、INTD 四种中断，软件可通过写 IOR: PCIE0/1_INTx_INDEX 指定这四中断中断对应的中断向量号，通过写 IOR: INT 中断配置寄存器进行配置，其中 INTx (INTA/B/C/D)、aerErrInt_CONFIG、pmeInt_CONFIG、hpInt_CONFIG、linkAutoBwInt_CONFIG、bwMgtInt_CONFIG 寄存器分别记录 INTx、高级错误报告错、PCI-E 电源管理、热插拔、链路自动带宽调节、链路带宽管理产生的 INT 中断信息，这些寄存器可指定从 PCI-E 设备传上来的每种中断是否使能以及向核心报告的中断子类型（可选择 INTA、INTB、INTC 或 INTD 之一），并指定处理该中断的目标核心号。

PCI-E 接口部件检测到 PCI-E 设备产生了 INTx 中断时，置对应的 IOR: INT 中断配置寄存器中的有效位为“1”，然后查找该寄存器的其它域信息，如果中断使能，则向系统接口部件发送对指定核心的 INTx 中断请求。

系统接口部件接到了 INTx 中断后，检查指定的目标核心状态，如果可处理中断且中断使能（IOR: INTEN 为“1”），则向该核心发 INTx 中断请求，并携带中断向量号；如果核心睡眠且配置设备中断需要唤醒核心，则设备中断丢失，在 IOR: DEVINT_MIS 中登记，同时唤醒核心；如果核心睡眠且配置设备中断不需要唤醒核心，或中断不使能（IOR: INTEN 为“0”），或核心断连，则设备中断丢失，在 IOR: DEVINT_MIS 中登记，同时通知维护接口发生了软件错，登记 IOR: SI_FAULT_STAT，可配置产生故障中断和向系统报错。

目标核心在处理该中断的程序中，需要通过读 IOR: INT 中断配置寄存器，将其中的 Valid 域清“0”，同时获得产生 INTx 中断的具体类型，并进行相应处理。

5.1.3.2 PCI-E 的 MSI 中断

每个 PCI-E 接口的 MSI 中断包括 256 个中断子类型, 软件可通过写 IOR: PCIE0/1_MSI[i]_INDX 指定各种中断对应的中断向量号。

软件写 IOR: MSIXAddr, 可以指定当 PCI-E 端设备对该地址进行 DMA 写操作时, 产生 MSI0~MSI255 中断请求, 请求中断信息记录在 MSI 中断信息 IOR: MSIX_CONFIG0~255 中; 除 DMA 写产生的 MSI 中断外, 高级错误报告中断、PCI-E 电源管理中断、热插拔中断产生的 MSI 中断信息分别记录在 MSI 中断信息 IOR: aerErrMsi_CONFIG、pmeMsi_CONFIG、hpMsi_CONFIG 中。软件可以通过 I/O 写来配置上述 MSI 中断信息寄存器, 这些寄存器可以指定从 PCI-E 接口传上来的每种中断是否使能以及向核心报告的中断子类型 (高级错误报告、PCI-E 电源管理、热插拔三类中断可选择 MSI0~MSI31 中断请求中的一种; 其他普通 MSI 中断可选择 MSI0~MSI255 中断请求中的一种), 并指定处理该中断的目标核心号。

硬件检测到 PCI-E 设备产生了 MSI 中断时, 置对应的 IOR: MSI 中断配置寄存器的有效位为“1”, 然后查找该寄存器的其它域信息, 如果中断使能, 则向系统接口部件发送对指定核心的 MSI 中断请求。

系统接口部件接到 MSI 中断后, 检查指定的目标核心状态, 如果可处理中断且中断使能 (IOR: INTEN 为“1”), 则向该核心发 MSI 中断请求, 并查询获得中断向量号; 如果核心睡眠且配置设备中断需要唤醒核心, 则设备中断丢失, 在 IOR: DEVINT_MIS 中登记, 同时唤醒核心; 如果核心睡眠且配置设备中断不需要唤醒核心, 或中断不使能 (IOR: INTEN 为“0”), 或核心断连, 则设备中断丢失, 在 IOR: DEVINT_MIS 中登记, 同时通知维护接口发生了软件错, 登记 IOR: SI_FAULT_STAT, 可配置产生故障中断和向系统报错。

目标核心在处理该中断的程序中, 软件还需要通过读 IOR: MSI 中断配置寄存器, 将其中的 Valid 域清“0”, 同时获得产生 MSI 中断的具体类型, 并进行相应处理。

5.1.3.3 短时钟中断

软件可通过写 IOR: SHORT_CLK_START, 指定短时钟计数器的初值; 通过写 IOR: SHORT_CLK_MSK, 控制不发送短时钟中断的核心; 通过写 IOR: SHORT_INDX, 指定短时钟中断对应的向量号。

系统接口中的短时钟计数器从指定的初值开始, 每个维护时钟周期执行减“1”操作, 当计数器的值减到“0”时, 便产生一个短时钟中断。系统接口按核组 (申威 410 相当于只有一个核组) 进行广播, 同一核组内进行轮转原则向核心转发短时钟中断, 核组内轮转时, 所有核心都参与轮转, 如果选择的核心不能处理中断或中断不使能, 则发向该核心的短时钟中断丢失; 否则向目标核心发短时钟中断, 并携带中断向量。

5.1.3.4 维护中断

软件可通过写 IOR: MT_INDEX, 指定维护中断对应的中断向量号。

当维护接口接到对 IOR: MT_INT 的维护 I/O 写命令, 且中断类型是普通维护中断时, 维护接口向系统接口发送一个对指定目标核心的维护中断请求。

系统接口接到该中断后, 检查指定的目标核心状态, 如果该核心可处理中断且中断使能 (IOR: INTEN 为“1”), 则将维护中断发向该核心, 并携带中断向量; 如果核心睡眠且配置设备中断需要唤醒核心且中断使能 (IOR: INTEN 为“1”), 则设备中断丢失, 在 IOR: DEVINT_MIS 中登记, 同时唤醒核心; 如果核心睡眠且配置设备中断不需要唤醒核心, 或中断不使能 (IOR: INTEN 为“0”), 或核心断连, 则设备中断丢失, 在 IOR: DEVINT_MIS 中登记, 同时通知维护接口发生了软件错, 登记 IOR: SI_FAULT_STAT, 可配置产生故障中断和向系统报错。

5.1.3.5 故障中断

当申威 411 处理器内各硬件发生故障时, 在本地记录详细的故障信息, 同时在维护接口部件的 IOR: CGx_FAULT_STAT、SI_FAULT_STAT 中汇集总的故障信息。软件可通过写 IOR: CGx_FAULT_INT_EN, SI_FAULT_INT_EN 指定那些故障发生时需要产生故障中断; 通过写 IOR: SYSERR_INDEX, 指定故障中断对应的向量号。

软件置 IOR: SOFT_INF[ERROR]为“1”时, 也会产生故障中断。

系统接口内可配置两种故障中断处理方式, 一种是轮转方式, 一种是指定核心方式。轮转方式下, 根据轮转指针依次检查核心是否处于可处理中断状态且中断使能 (IOR: INTEN 为“1”) 状态, 如果不满足条件, 则跳过该核心, 如果所有核心都不满足条件, 则故障中断丢失, 在 IOR: DEVINT_MIS 中登记, 同时通知维护接口发生了软件错, 登记 IOR: SI_FAULT_STAT, 可配置向系统报错, 但不产生故障中断。

指定核心方式下, 如果目标核心处于可处理中断状态且中断使能 (IOR: INTEN 为“1”), 则向该核心发故障中断, 并携带中断向量号; 如果目标核心睡眠, 或中断不使能 (IOR: INTEN 为“0”), 或断连, 在其它可接收中断的核心中进行轮转, 处理方式与轮转方式相同。

5.1.4 特殊中断

5.1.4.1 睡眠中断

睡眠中断是一种特殊中断, 目的是为了在核心无工作负载时降低运行功耗。核心只能在非用户

模式下向本核心或其它核心发出睡眠中断。维护命令也可以向指定核心发送睡眠中断。

软件可通过写核心 CSR: INT_VEC[SLEEP_VEC], 指定睡眠中断对应的中断向量号。当软件写 CSR: INT_REQ 时, 如果指定的中断类型为睡眠中断, 则源核心向系统接口部件发送一个或多个核心睡眠中断。当维护接口接到一个对 IOR: MT_INT 的维护写命令, 并且中断类型是睡眠中断时, 维护接口也向系统接口发起一个对指定核心的睡眠中断。

系统接口接到一个睡眠中断后, 如果指定核心可处理中断状态且中断使能(IOR: INTEN 为“1”), 则将睡眠中断发向该核心; 否则该中断丢失, 并在 IOR: INTLOS 中相应的位置进行登记。

目标核心接到睡眠中断请求后, 如果 CSR: IER0~3 中对应的睡眠中断使能, 则硬件自动在 CSR: INT_STAT0~3 中登记, 登记的位置由 CSR: INT_VEC[SLEEP_VEC]决定, 并进入睡眠中断处理特权程序入口 (SLEEP) 进行处理。当睡眠中断服务程序保留完现场后, 通过写中断目标核心的 CSR: INT_CLR0~3 中对应位, 硬件将自动清除目标核心的 CSR: INT_STAT0~3 中的对应位。随后写 IOR: INTEN, 关闭对该目标核心的中断使能。最后写 IOR: SLEEP_DONE_x, 通知维护接口某个核心可以睡眠, 以后中断目标核心的时钟被降为最低频率, 并一直处于复位状态, 不响应睡眠唤醒和复位以外的任何中断。

注意: 申威 411 处理器可能将不同源核心发向同一个目标核心的睡眠中断合并。另外, 硬件保证, 当睡眠中断到达目标核心时, 如果对应的中断使能没有打开 (即 CSR: II_ER[II_EN]为“0”), 则保留该中断, 直到被软件清除为止。

5.1.4.2 睡眠唤醒中断

睡眠唤醒中断是一种特殊中断, 使得处于睡眠状态的核心恢复正常运行, 核心只能在非用户模式下发出对其它核心的睡眠唤醒中断, 维护命令也可以对指定核心发送睡眠唤醒中断。

当软件写 CSR: INT_REQ 时, 如果指定的中断类型是睡眠唤醒中断, 则硬件向系统接口发出一个或多个核心睡眠唤醒中断。系统接口部件收到睡眠唤醒中断后, 直接将睡眠唤醒中断发向维护接口部件。

当维护接口收到一个对 IOR: MT_INT 的维护 I/O 写命令, 并且类型是睡眠唤醒中断时, 如果该核心处于连接状态 (IOR: CORE_ONLINE 为“1”)且已经在睡眠, 则直接唤醒指定核心。

维护接口接到唤醒请求时, 如果核心没有断开, 则先将目标核心的工作时钟恢复, 然后置目标核心的复位信号无效, 并根据 IOR: INIT_CTL[HOTRST_CTL]选择进行存储器自测试的方式, 随后让目标核心进入睡眠唤醒中断处理特权程序入口 (WAKEUP) 执行。目标核心响应睡眠唤醒中断后, 在睡眠中断处理程序中, 软件通过写 IOR: INTEN, 打开该核心的使能位。

对没有睡眠的核心发送睡眠唤醒中断, 不产生任何效果。需要注意的是, 申威 411 处理器可能将不同源核心发向同一个目标核心的睡眠唤醒中断合并。

5.1.4.3 复位

复位是一种特殊的中断,包括由引脚信号 `DCOK_H` 引起的上电复位和由引脚信号 `Reset_L` 引起的冷复位,都是电平触发的中断。当硬件监测到这两个信号有效时,对申威 411 处理器进行完全复位。当硬件监测到这两个信号无效时,申威 411 处理器进入初始化流程,根据配置进行存储器自测试和指令 Cache 加载,最后进入复位特权程序入口 (`RESET`) 开始取指令运行。复位相关的处理参看第 7 章。

5.2 中断的优先级

各种中断特权程序的入口地址参看 2.2.2,中断的优先级从低到高依次如下(当多个中断请求同时发生时,硬件先进入优先级高的中断特权程序入口):

- 1) 一般中断 0,即核间中断;
- 2) 一般中断 1,包含事件计数中断、已纠正错中断、外部硬件中断;
- 3) 睡眠中断;
- 4) 机器检查错中断;
- 5) 复位和睡眠唤醒中断。

对于一般中断 1,申威 411 处理器硬件不设置具体的中断优先级,由软件根据需要设置中断优先级。具体设置方法可通过设置对 `CSR: INT_STAT0~3` 的检查顺序和设置 `IER0~3` 的值来实现。首先写 `CSR: IER0~3` 开放所有使能的中断,若某中断被响应并进入到中断特权程序入口,则保存当前的中断使能寄存器 `CSR: IER0~3`,并按定义的优先级从高到低依次查询 `CSR: INT_STAT0~3` 中的相应位,若检测到某中断被置位,则不再查询 `CSR: INT_STAT0~3` 中较低优先级的中断位,将指令流转到该中断对应的处理程序。在进入到相应中断的处理程序之前,保留原中断使能寄存器 `CSR: IER0~3`,修改 `IER0~3`,关闭与当前处理的中断优先级相同或更低的中断使能,开放比本中断优先级更高的中断。在当前中断处理结束并退出之前,用保存的值恢复中断使能寄存器 `CSR: IER0~3`。

5.3 中断的嵌套

申威 411 处理器可通过 `HM` 模式来实现多级中断的嵌套。当核心响应中断请求时,先进入 `HM` 模式,由于 `HM` 模式下不响应复位以外的任何中断请求,因此可以确保中断现场不被破坏。软件可先读取当前中断源信息,并将与当前中断同级的中断或比当前中断级别更低的中断屏蔽,然后转入非 `HM` 模式,执行真正的中断处理程序。在中断处理过程中,允许更高级别的中断中止当前中断的处理,进入更高优先级的中断处理,从而实现中断的嵌套。

中断总是在指令的边界上被识别，且申威 411 处理器的核心保证断点之前的指令都无异常完成并退出。当中断请求中断当前程序的处理时，硬件自动将断点处的指令地址保存到 CSR: EXC_PC 中，以便中断处理结束后能正确返回到断点处继续执行。

6 异常

异常是指在指令处理的过程中产生的、与指令流同步的、由程序或软件原因而引起的问题，包括两种类型：

1) 故障 (Fault): 表示指令或指令的操作非法，在指令处理过程中产生，产生故障的指令并未正常退出，而是被中止 (Abort)；

2) 算术自陷 (Arithmetic Trap): 表示运算类指令在运算时，操作数或运算结果出现异常，产生算术自陷的指令都将处理完毕并正常退出。

申威 411 处理器为两种异常保留的断点都是精确的，并将发生异常的指令地址记录在 CSR: EXC_PC 中。

6.1 DTB 脱靶

当数据流访问 DTB 时，如果既不命中一级 DTB，也不命中二级 DTB，则产生 DTB 脱靶故障，判断 DTB 命中的方法参看 4.4.6 节。当产生 DTB 二次脱靶时，进入 DTBM_DOUBLE 特权程序入口。当发生 DTB 一次脱靶时，如果数据流访问虚地址的[42]位为“0”，则进入 DTBM_SINGLE_0 特权程序入口，否则进入 DTBM_SINGLE_1 特权程序入口。

DTB 一次脱靶故障 (DTBM_SINGLE_x) 的处理一般需要装填 DTB，但首先要分析故障的数据流地址，确定地址指示的存储空间是否允许访问。当发生 DTB 一次脱靶时，硬件自动在 CSR: EXC_PC 中存放异常指令的 PC 值，在 CSR: DVA 中存放异常指令访问的数据流虚地址，在 CSR: DS_STAT[OPCODE(5:0)]中存放异常指令的操作码，在 CSR: DS_STAT[WR]中存放异常指令的写标志，在 CSR: EXC_SUM[REG(4:0)]中存放异常指令的 Ra 域。软件可通过读 CSR: DVA_FORM 同时获得虚页表基址和虚页号等信息，提高处理效率。DTB 一次脱靶故障的处理过程中可能会引起 DTB 二次脱靶故障。

DTB 二次脱靶故障 (DTBM_DOUBLE) 是在处理 ITB 脱靶或 DTB 一次脱靶时产生，指示需要操作系统按多级页表来获得正确的物理地址页面号，引起这种故障的指令必定是 PRI_LDx/vpte，此时 CSR: DS_STAT 和 CSR: DVA 内容维持不变，只在 CSR: EXC_PC 中存放引起二次脱靶指令的 PC 值，在 CSR: EXC_SUM[REG(4:0)]中该指令的 Ra 域。

6.2 ITB 脱靶

当指令流访问 ITB 时，如果不命中 ITB，则产生 ITB 脱靶故障，判断 ITB 命中的方法参看 4.4.5。当发生 ITB 脱靶时，如果指令流访问虚地址[42]位为“0”，则进入 ITBM_SINGLE_0 特权程序入口，

否则进入 ITBM_SINGLE_1 特权程序入口。

ITB 脱靶故障 (ITBM_SINGLE_x) 的处理一般需要装填 ITB, 但首先要分析故障的指令流地址, 确定地址指示的存储空间是否允许访问。当发生 ITB 脱靶时, 硬件自动在 CSR: EXC_PC 中记录发生异常指令的 PC 值。

软件可通过读 CSR: IVA_FORM 同时获得虚页表基址和虚页号等信息, 提高处理效率。ITB 脱靶故障的处理过程中, 可能会引起 DTB 二次脱靶故障。

6.3 浮点屏蔽故障

当核心在 CSR: UPCR[FPE]和 CSR: UPCR[FCE]中任意一位为“0”时执行浮点指令或 SIMD 扩展指令, 或者在 CSR: UPCR[SCE]为“0”时执行 SIMD 扩展指令, 都将产生浮点屏蔽故障 (FEN)。只有合法且非空指令才会产生浮点屏蔽故障, 某些目标寄存器为 F31 的 FLDS 和 FLDD 指令不产生浮点屏蔽故障。当产生浮点屏蔽故障时, 硬件自动在 CSR: EXC_PC 中保存异常指令的 PC 值, 在 CSR: IGL_INST 中记录异常指令的 32 位信息, 并进入 FEN 特权程序入口。

6.4 不对界故障

申威 411 处理器对访存类指令进行地址检查时, 如果地址不对界, 则产生不对界故障 (UNALIGN), 硬件自动在 CSR: EXC_PC 中记录异常指令的 PC 值, 在 CSR: DVA 中记录有效的数据流访问地址, 在 CSR: DS_STAT[OPCODE(5:0)]中存放异常指令的操作码, 在 CSR: DS_STAT[WR]中存放异常指令的写标志, 在 CSR: EXC_SUM[REG(4:0)]中存放异常指令的 Ra 域, 然后进入 UNALIGN 特权程序入口。地址不对界检查的方法参看 4.4.3。

6.5 数据流故障

数据流故障 (DFAULT) 是对访存指令产生的数据流地址进行地址检查或进行虚实地址转换时产生, 数据流故障的类型很多, 参看 4.4.3、4.4.6、4.5.3。产生数据流故障时, 硬件自动在 CSR: EXC_PC 中记录该访存指令的 PC 值, 在 CSR: DVA 中记录有效的数据流访问地址, 在 CSR: DS_STAT[OPCODE(5:0)]中存放异常指令的操作码, 在 CSR: DS_STAT[WR]中存放异常指令的写标志, 在 CSR: EXC_SUM[REG(4:0)]中存放异常指令的 Ra 域, 然后进入 DFAULT 特权程序入口。各种数据流故障类型与 CSR: DS_STAT 中标志位的对应关系如下:

- 1) 数据流虚地址符号扩展错, 此时 CSR: DS_STAT[BAD_DVA]为“1”;
- 2) 数据流地址匹配故障, 此时 CSR: DS_STAT[DA_MATCH]为“1”;
- 3) DTB 虚实地址转换时发生访问越权故障, 此时 CSR: DS_STAT[ACV0]为“1”;
- 4) DTB 虚实地址转换时发生读访问故障, 此时 CSR: DS_STAT[FOR]为“1”;

- 5) DTB 虚实地址转换时发生写访问故障, 此时 CSR: DS_STAT[FOW]为“1”;
- 6) 读写访问可 Cache 空间时, 或读访问不可 Cache 空间时, 数据流访问地址超出申威 411 处理器主存容量, 或读访问不存在的 IOR, 此时 CSR: DS_STAT[ACV1]为“1”;
- 7) SIMD 扩展访存指令、原子操作指令、不可 Cache 访存指令访问 I/O 空间地址, 或锁指令访问 I/O 空间或不可 Cache 空间, 此时 CSR: DS_STAT[ACV1]为“1”;
- 8) 数据 Cache 发生 DTAG 偶校验错故障, 此时 CSR: DS_STAT[TAG_PERR]为“1”;
- 9) Load 指令访问数据 Cache, 发生数据 ECC 错故障, 此时 CSR: DS_STAT [DAT_ERR]为“1”;
- 10) 对可 Cache 存储器空间的 load 类及 store 类访问, 以及对不可 Cache 空间和 I/O 空间的 load 类访问时, 返回给的响应中带 ECC 多错, 此时 DS_STAT[MEM_MERR] 为“1”。

6.6 指令流故障

指令流故障 (IACV) 是指令流地址进行地址检查或进行虚实地址转换时产生的, 产生指令流故障的类型很多, 参看 4.4.3 和 4.4.5。产生指令流故障时, 硬件自动在 CSR: EXC_PC 中记录异常指令的 PC 值, 然后进入 IACV 特权程序入口。各种指令流故障类型与 CSR 中登记的信息对应关系如下:

- 1) ITB 虚实地址转换时发生访问越权故障, 此时以 CSR: EXC_SUM[BAD_IVA、IA_MATCH、IDA_MATCH]为全“0”来表示;
- 2) 指令流地址匹配成功, 此时 CSR: EXC_SUM[IA_MATCH]为“1”;
- 3) 跳转目标指令地址发生符号扩展错, 此时 CSR: EXC_SUM[BAD_IVA]为“1”, 并在 CSR: IVA 中登记跳转的目标指令地址;
- 4) 整数转移目标指令地址匹配成功, 此时 CSR: EXC_SUM[IDA_MATCH]为“1”, 并在 CSR: IVA 中登记跳转的目标指令地址。

6.7 操作码非法

指令流水线的译码站台对指令的操作码和功能码进行检查, 遇到以下情况时将产生操作码非法故障 (POCDEC):

- 1) 指令的操作码为保留的操作码;
- 2) 指令的功能码为保留的功能码;
- 3) SYS_CALL 指令的功能码非法, 具体检查方法参看 2.2.1;
- 4) 在非 HM 模式下使用了特权指令;
- 5) CSR:CP_CTL[EGB]配置为 0 时, 译码出 EVICTDG 指令;
- 6) CSR:CP_CTL[ELB]配置为 0 时, 译码出 EVICTDL 和 FLUSHD 指令;

7) 在硬件模式下遇到 Halt 指令;

8) 在结构手册中已定义,但在芯片中没有实现的指令;

9) 锁指令没有按要求成对出现在指定位置,具体为:从 ICache 中取出对界的 4 条指令,如果 LSTx 指令不在指令 0 位置(或指令 2 位置),或者指令 1 位置上(或指令 3 位置)不是非空的 RD_F 指令,则 LSTx 指令报操作码非法故障;如果指令 1 位置上(或指令 3 位置)是非空的 RD_F 指令,但指令 0 位置(或指令 2 位置)上不是 LSTx 指令,或者非空的 RD_F 指令不在指令 1 位置(或指令 3 位置),则 RD_F 指令报操作码非法故障。

当发生操作码非法故障时,硬件自动在 CSR: EXC_PC 中记录异常指令的 PC 值,在 CSR: IGL_INST 中记录异常指令的 32 位信息,然后进入 OPCDEC 特权程序入口。当锁指令没有按要求成对出现在指定位置时,硬件自动置 CSR: IGL_INST[LOCK_IGL]为“1”。

6.8 算术自陷

整数和浮点运算(包括整数和浮点的数据转换)以及原子操作指令在执行过程中,如果产生了异常并且没有被屏蔽,则称之为算术自陷。共有 7 种算术自陷,其中整数溢出在整数运算和浮点到整数的转换中都可能产生,原子操作溢出只可能在原子操作指令执行时产生,其它 5 种算术自陷只可能在浮点运算中产生。

发生算术异常时,异常的指令总是可以完成处理并退出,如果对应的自陷使能,则硬件自动在 CSR: EXC_PC 中存放异常指令的 PC 值,在 CSR: EXC_SUM[REG(4:0)]中存放异常指令的目标寄存器域,并在 CSR: EXC_SUM 中登记算术自陷类型,如果 FPCR[EXC_CTL(1)]为“0”,则要设置 CSR: EXC_SUM[SWC]为“1”,然后进入 ARITH 特权程序入口;若对应的自陷不使能则硬件继续执行。

发生浮点算术异常时,无论对应的自陷是否使能,硬件都要自动在 FPCR 中设置该异常对应的状态信息,具体参考《申威 411 处理器程序员手册》。

6.8.1 整数溢出

整数溢出(OVI)表示整数运算指令和浮点到整数的转换指令执行过程中,产生的结果超出了目标寄存器的最大表示范围。进入整数溢出算术自陷的情况如下:

1) 整数运算指令产生整数溢出且 CSR: ICR[FD]为“0”时,需设置 CSR: EXC_SUM[OVI]为“1”,设置 CSR: EXC_SUM[INT]为“1”;

2) 浮点转换指令产生整数溢出且 FPCR[EXC_CTL(1:0)]为“00₂”时,需设置 CSR: EXC_SUM[OVI]为“1”,设置 CSR: EXC_SUM[INT]为“0”;如果 FPCR[INED]为“0”,则还要设置 CSR: EXC_SUM[INE]为“1”;

3) 浮点转换指令产生整数溢出且 FPCR[EXC_CTL(1:0)]为“01₂”或“10₂”时,需设置 CSR:

EXC_SUM[OVI]为“1”，设置 CSR: EXC_SUM[INT]为“0”；

当浮点指令发生整数溢出时，无论整数自陷是否屏蔽，都要设置 FPCR[OVI]为“1”，设置 FPCR[INE]为“1”。

6.8.2 无效操作

执行浮点运算类指令时，如果当前操作的操作数存在非规格化数（Denormal）或对当前的操作而言是无效数据，则可能产生无效操作异常（INV）。产生无效操作算术自陷的各种情况如下：

- 1) FPCR[EXC_CTL][1]为“1”时产生无效操作异常，则设置 CSR: EXC_SUM[INV]为“1”；
- 2) FPCR[EXC_CTL][1]为“0”，操作数含非规格化数（Denormal）且 DNZ 为“0”，此时必定产生无效操作异常，设置 CSR: EXC_SUM[INV]为“1”；

3) FPCR[EXC_CTL][1]为“0、INVD 为“0”且操作数不含非规格化数（Denormal）时产生了无效操作异常，则设置 CSR: EXC_SUM[INV]为“1”；

4) FPCR[EXC_CTL][1]为“0”、INVD 为“0”、操作数含非规格化数（Denormal）且 DNZ 为“1”时，如果将 Denormal 数当“0”处理后仍然产生无效操作异常，则设置 CSR: EXC_SUM[INV]为“1”。

其中有两种情况被当作 Denormal 自陷，硬件不自动修改 FPCR，其它情况无论无效操作自陷是否屏蔽都要设置 FPCR[INV]为“1”。作为 Denormal 自陷的情况如下：

- 1) FPCR[EXC_CTL][1]为“1”且操作数含 Denormal 数；
- 2) FPCR[EXC_CTL][1]为“0”、操作数含 Denormal 数且 DNZ 为“0”。

6.8.3 下溢

浮点运算指令执行过程中，如果舍入结果的幅值（绝对值）小于目标数据格式的最小有限值，则产生下溢（UNF）。产生下溢算术自陷的各种情况如下：

1) FPCR[EXC_CTL(1:0)]为“00₂”且 FPCR[UNFD]为“0”或 FPCR[UNDZ]为“0”时产生下溢异常，则设置 CSR: EXC_SUM[UNF]为“1”；如果 FPCR[INED]为“0”，还要设置 CSR: EXC_SUM[INE]为“0”；

2) FPCR[EXC_CTL(1:0)]为“01₂”且 FPCR[UNFD]为“0”或 FPCR[UNDZ]为“0”时产生下溢异常，则设置 CSR: EXC_SUM[UNF]为“1”；

3) FPCR[EXC_CTL(1:0)]为“10”时产生下溢异常，则设置 CSR: EXC_SUM[UNF]为“1”。

无论下溢自陷是否屏蔽，都要设置 FPCR[UNF]为“1”，设置 FPCR[INE]为“1”。

6.8.4 上溢

浮点运算指令执行过程中，如果舍入结果的幅值（绝对值）超过目标格式的最大有限值时即产

生上溢 (OVF)。产生上溢算术自陷的各种情况如下:

- 1) FPCR[EXC_CTL(1:0)]为“00₂”且 FPCR[OVFD]为“0”时产生上溢异常, 则设置 CSR: EXC_SUM[OVF]为“1”; 如果 FPCR[INED]为“0”, 还要设置 CSR: EXC_SUM[INED]为“0”;
 - 2) FPCR[EXC_CTL(1:0)]为“01₂”且 FPCR[OVFD]为“0”时产生上溢异常, 则设置 CSR: EXC_SUM[OVF]为“1”;
 - 3) FPCR[EXC_CTL(1:0)]为“10₂”时产生上溢异常, 则设置 CSR: EXC_SUM[OVF]为“1”。
- 无论上溢自陷是否屏蔽, 都要设置 FPCR[OVF]为“1”, 设置 FPCR[INE]为“1”。

6.8.5 除数为零

浮点除法指令的除数为“0”, 而被除数为有限非“0”数, 则产生除数为零异常 (DZE)。产生除数为零算术自陷的各种情况如下:

- 1) FPCR[EXC_CTL1]为“1”, 操作数不含非规格化数 (Denormal), 如果产生除数为零异常, 则设置 CSR: EXC_SUM[DZE]为“1”;
- 2) FPCR[EXC_CTL1]为“0”且 DZED 为“0”, 操作数不含非规格化数 (Denormal), 如果产生除数为零异常, 则设置 CSR: EXC_SUM[DZE]为“1”;
- 3) FPCR[EXC_CTL1]为“0”且 DZED 为“0”, 操作数含非规格化数 (Denormal) 且 DNZ 为“1”, 如果产生除数为零异常, 则设置 CSR: EXC_SUM[DZE]为“1”。

发生非规格化数自陷 (Denormal 自陷) 时, 硬件不自动修改 FPCR, 其它情况无论自陷是否屏蔽都要设置 FPCR[DZE]为“1”。

6.8.6 非精确结果

浮点运算指令执行过程中, 发生下列任一情况, 则产生非精确结果异常 (INE):

- 1) 舍入后的结果与舍入前的真值不一致;
- 2) 发生整数溢出或舍入时产生上溢 (不论上溢自陷是否屏蔽);
- 3) 舍入时产生下溢 (不论下溢自陷是否屏蔽)。

舍入后的结果仍将保留到目标寄存器中。如果 FPCR[EXC_CTL(1:0)]为“00₂”, 且 FPCR[INED]为“0”, 则进入算术自陷, 并设置 CSR: EXC_SUM[INE]为“1”。无论自陷是否屏蔽, 都要设置 FPCR[INE]为“1”。

6.8.7 原子操作溢出

非特权原子操作指令在执行过程中, 运算结果超出了目标寄存器的最大表示范围, 则产生原子操作溢出 (INCO), 如果 CSR: ICR[FD]为“0”且 CSR: DC_CTL[ATOP_OVF_EN]为“1”, 则产

生算术自陷，设置 CSR: EXC_SUM[INCO]为“1”。

7 复位和初始化

7.1 复位的种类

申威 411 处理器支持多种复位，包括上电复位、冷复位和睡眠复位（或称为核心复位），不同的复位由不同的事件引起，申威 411 处理器内部的操作也不完全相同，但复位结束后都从复位的特权程序入口开始取指令运行。表 7-1 列出各种不同复位的主要差别。

表 7-1: 各种复位的差别

复位类型	上电复位	冷复位	睡眠与唤醒
触发事件	外部触发 DCOK_H 信号无效	外部触发 Reset_L 信号有效	开始复位：写睡眠寄存器 结束复位：写睡眠唤醒寄存器
范围	申威 411 处理器	申威 411 处理器	指定核心
锁相环 PLL 时钟频率	上升	先下降，再上升	指定核心的时钟降频为维护 时钟的 8 分频，核心唤醒后恢 复为正常频率
BISR 自修复	根据配置	根据配置	根据配置
BIST 自测试	可选择其中一种 运行	可选择其中一种 运行	可选择其中一种 运行
指令 Cache 加载	加载初始化程序	加载初始化程序	不加载
CSR: PRI_BASE	[38:37]为核心组编号，其 它位清“0”	[38:37]为核心组编号，其 它位清“0”	保持不变
CSR: EXC_PC	不变	不变	睡眠中断断点的 PC 值
CSR: INT_STAT0~ 3	清“0”	清“0”	保持不变
CSR: INT_VEC	设置初值	设置初值	保持不变
CSR: IIO	清“0”	清“0”	保持不变
CSR: IIO1	清“0”	清“0”	保持不变
CSR:	清“0”	清“0”	保持不变

II_MAIL			
CSR: ISSUE_CTL	设置为 0xF,FFFF	设置为 0xF,FFFF	保持不变
复位后 工作频率	重新设置	不变	不变

7.2 上电复位

上电复位从系统对申威 411 处理器加电开始，并维持引脚信号 DCOK_H 无效和 Reset_L 有效，使申威 411 处理器内部的所有状态被清除。待电源稳定、输入给申威 411 处理器的时钟就绪之后，系统使 DCOK_H 有效，申威 411 处理器内部的 PLL 开始升频。当 PLL 达到配置频率、处于稳定工作状态后，系统使 Reset_L 无效，结束上电复位。复位结束后，等待系统通过维护接口配置申威 411 处理器内部的相关寄存器，并根据配置对内部存储器进行自修复（BISR）、自测试（BIST）或初始化（BISI），再根据配置将初始化程序加载到各核心的指令 Cache 或申威 411 处理器控制的存储器中，最后启动各核心开始运行。

初始化程序主要包含以下内容：

- 1) 初始化 31 个整数寄存器、31 个浮点寄存器和 31 个向量寄存器；
- 2) 初始化核心内的控制与状态寄存器（CSR）；
- 3) 读 DDR3 存储器接口相关的 I/O 寄存器，判断是否需要通过软件进行 DDR3 存储器接口的数据训练（Data Training）来设置各类接口参数。如果需要进行数据训练，则选择一个核心进行数据训练，其它核心等待数据训练完成；
- 4) 所有核心参与进行存储器测试；
- 5) 存储器测试完成后，将全部存储器空间清“0”，读取存储器中的标志确定进一步操作。

系统维护将操作系统需要的文件加载到申威 411 处理器控制的存储器中，并在存储器中设置必要的标志，通知所有核心跳转到特权程序入口（RESET）开始取指令运行。

上电复位一般用于系统加电后的复位，在需要改变申威 411 处理器的工作频率时，也需要采用上电复位，但这种情况需要外部系统的支持，因为申威 411 处理器的工作频率与申威 411 处理器的配置参数密切相关，对应的初始化程序也可能不同。

7.3 冷复位

冷复位是在 DCOK_H 信号维持有效的前提下，由信号 Reset_L 来实现的。Reset_L 有效，使申威 411 处理器进入冷复位，清除内部状态，复位期间，不影响内部 PLL 的正常工作。Reset_L 无效后，结束冷复位，此后的处理与上电复位相同。

7.4 睡眠复位

睡眠复位是一种由申威 411 处理器自身或外部其它核心触发的复位形式，可使核心经过睡眠复位进入睡眠状态，核心时钟可以降到很低的频率，从而降低核心的运行功耗。睡眠的核心被唤醒后，又自动返回到正常工作状态。进入睡眠状态和从睡眠状态唤醒，都需要操作系统和申威 411 处理器外部系统的支持。

7.4.1 睡眠流程

申威 411 处理器接收到睡眠中断请求后，进入睡眠中断服务程序，开始核心睡眠前的准备工作，具体操作流程如下：

- 1) 操作系统根据需要，进行现场保留，保留的内容包括：
 - a. 将所有必要的 CSR 寄存器内容保存到主存；
 - b. 将所有整数与浮点寄存器内容保存到主存；
 - c. 将其它必要的信息保存到主存；
 - d. 刷新所有 Cache（包括指令 Cache、数据 Cache 和二级 Cache）。
- 2) 写 IOR: CPM_CoreSleep（软件可以写任意值，硬件遇到此 IOR 的写请求时总是写入“1”，并在核心被唤醒时隐式清“0”），通知核外禁止向即将进入睡眠状态的核心发送查询请求；
- 3) 查询 IOR: CPM_ProbeOK 是否为“1”，确认先前发向对该核心的查询请求均已返回回答；
- 4) 最后写: SLEEP_DONE_x 为“1”，表示对应核心可以进入睡眠模式。
- 5) 维护接口启动该核心进入睡眠状态并使其处于复位状态，且工作时钟频率降为最低频率。

7.4.2 唤醒流程

申威 411 处理器的核心从睡眠状态唤醒的方式有以下几种：

1) 处于非睡眠状态的核心通过写 CSR: INT_REQ，且指定[IntType]为 3，即可向指定核心发送睡眠唤醒中断请求；

2) 外部系统通过维护接口向指定核心发维护睡眠唤醒中断命令。

核心睡眠唤醒流程如下：

1) 睡眠唤醒中断由申威 411 处理器内部的维护接口统一处理。维护接口接收到唤醒中断请求后，检查中断的目标核心是否处于睡眠状态，如果是且处于非断连状态，则启动唤醒流程，将睡眠核心的时钟恢复到正常工作状态；

2) 被唤醒的核心结束复位状态，根据 IOR: INIT_CTL[HOTRST_CTL]可选择进行内部初始化（BISI）或自修复（BISR），清除有关状态和各 Cache 内容，然后从睡眠唤醒的特权程序入口（由

于睡眠复位不清除 CSR: PRI_BASE 内容, 可以与上电复位和冷复位区分开来) 处取指令执行;

3) 复位的特权程序读取保存在主存中的现场信息, 恢复 CSR、整数和浮点寄存器及其它现场信息, 然后返回进入睡眠前的状态。

睡眠唤醒过程中对指令 Cache 进行了初始化处理, 也没有向指令 Cache 加载初始化程序, 因此会从主存直接取指执行。

7.5 复位后的 CSR

申威 411 处理器在复位时, 大多数 CSR 内容没有进行初始化, 保留复位前的状态, 由复位的特权程序进行必要的处理, 还有一部分则由硬件自动进行复位初始化。有些 CSR 的复位与具体的复位类型有关。三种复位对 CSR 初始设置相同的情况如下:

1) 指令部件中的 CSR:

- a. CSR: IS_CTL, 复位时置为 “0x10088”;
- b. CSR: VPCR, 复位时置为全 “0”;
- c. CSR: UPCR, 复位时置为 “0x07”;
- d. CSR: IA_MATCH, 复位时置为全 “0”;
- e. CSR: IA_MASK, 复位时置为 “0x7,FFFF,FFFF,FFFF”;
- f. CSR: HSERR_CNT, 复位时置为全 “0”;
- g. CSR: HSERR_TH, 复位时置为 “0x1F”;
- h. CSR: IGL_INST, 复位时置为全 “0”;
- i. CSR: SOFT_ERR, 复位时置为全 “0”;
- j. CSR: TIMER_CTL, 复位时置为全 “0”;
- k. CSR: TIMER_TH, 复位时置为全 “0”;

2) 数据 Cache 管理部件中的 CSR:

- a. CSR: DS_CTL, 复位时置为全 “0”;
- b. CSR: DC_CTL, 复位时置为 “0xF”;
- c. CSR: DA_MATCH, 复位时置为全 “0”;
- d. CSR: DA_MASK, 复位时置为 “0x7FF,FFFF,FFFF”。

3) 二级 Cache 管理部件中的 CSR:

- a. CSR: SC_CTL, 复位时置为 “0x8FF”;
- b. CSR: INT_REQ, 复位时置为全 “0”;
- c. CSR: CP_CTL, 复位时置为 “8'b10011100”;
- d. CSR: MERGE_OVTIME, 复位时置为 “0x100”。

4) 整数部件中的 CSR:

- a. CSR: ICR, 复位时置为 “0”;

- b. CSR: IDA_MATCH, 复位时置为全“0”;
- c. CSR: IDA_MASK, 复位时置为“0x7FF,FFFF,FFFF”;
- d. CSR: TC, 复位时置为全“0”;
- e. CSR: TC_CTL, 复位时置为“0”。

需要软件在复位的特权程序中进行初始化的 CSR 如下:

- 1) 需要进行设置的 CSR 寄存器有: ITB_IA、VPT_BASE、IER0~3、II_ER、DTB_IA、DTB_PCR、CID[63:8];
- 2) 需要清除的 CSR 寄存器有: IS_STAT、PC0_CR、PC1_CR、DS_STAT、DC_STAT、SC_STAT。

8 事件计数

事件计数是对程序运行过程中发生的特定事件进行统计，给用户程序运行信息，以达到调试、修改、优化程序，提高程序执行效率的目的。

申威 411 处理器设置 2 个事件计数器，在相关 CSR 的控制下，实现对申威 411 处理器内部各种事件的计数，计数器溢出可产生事件计数中断。计数器的初值、计数使能控制、计数的事件选择都通过 CSR 来控制。系统通过事件计数中断服务程序，实现事件计数功能。

8.1 事件计数相关 CSR

下表是与事件计数相关的 CSR。

表 8-1: 与事件计数相关的 CSR

序号	名称	相关的域	含义
1	IER0~3	PCEN[1:0]	事件计数中断使能
2	INT_STAT0~3	PC[1:0]	事件计数中断请求
3	IS_CTL	PC_CM[1:0]	指定进行事件计数的处理器模式
		CM_PCE	在指定处理器模式下事件计数使能
		AM_PCE	在任何处理器模式下事件计数使能
		PC0_EN	事件计数器 0 使能
		PC1_EN	事件计数器 1 使能
4	PC0_CR	PC0_SEL[3:0]	选择事件计数器 0 的计数事件
		PC0[57:0]	事件计数器 0，可设置初值
5	PC1_CR	PC1_SEL[5:0]	选择事件计数器 1 的计数事件
		PC1[57:0]	事件计数器 1，可设置初值

8.2 事件计数的模式

事件计数分成 2 类，一是所有处理器模式下的事件计数，即在任何模式下都对所选事件进行计数；二是指定处理器模式下的事件计数，即在指定处理器模式下对所选事件进行计数。两者的控制有所不同，前者的事件计数使能由 CSR: IS_CTL[AM_PCE] 决定，后者则由 CSR: IS_CTL[CM_PCE] 决定。一般两类事件计数只允许一种事件计数使能，否则事件计数的结果不能正确反映程序运行的情况。

申威 411 处理器设置了两个事件计数器（PC0 和 PC1），分别受各自的使能信号控制，可以单独计数，也可以同时计数。

当事件计数使能（即 IS_CTL[PC0_EN]或 IS_CTL[PC1_EN]有效，且 IS_CTL[CM_PCE]或 IS_CTL[AM_PCE]有效）时，事件计数器 0 或 1 即对所选择的事件进行计数。计数器的初值可通过写 PC0_CR 和 PC1_CR 进行设置。当某事件计数器计数溢出时，若事件计数中断使能（即 CSR:IER0~3[PCEN(1:0)]有效），则中断摘要寄存器 INT_STAT0~3 [PC(1:0)]的对应位为“1”，触发对应的事件计数中断。事件计数器在溢出之后将从零开始继续计数，需要事件计数中断服务程序重新设置初值。计数器的具体数值可在中断服务程序中通过读 CSR: PC0_CR 和 PC1_CR 来获得。

事件计数器 0 的计数事件由 CSR: PC0_CR[PC0_SEL(3:0)]决定，具体定义如表 8-2。

表 8-2: 事件计数器 0 的计数事件

PC0_SEL[3:0]	事件计数器 0 的计数事件
0x0	完成处理并退出的指令计数（不包含空指令和 MEMB 指令）。
0x1	完成处理并退出的访存指令计数（含 EVICTD _x 、FETCHD _x 、FLUSHD 等指令）。
0x2	完成处理并退出的读访存指令计数（含 FETCHD 和 FETCHD_E 指令）。
0x3	执行站台执行的转移指令计数。
0x4	执行站台执行的条件转移指令计数。
0x5	执行站台执行的无条件转移指令计数。
0x6	执行站台执行的 CALL 和 JSR 指令计数。
0x7	执行站台执行的 RET 指令计数。
0x8	处理器周期计数。
0x9	ITB 访问次数计数。
0xA	DTB 访问次数计数。
0xB	指令 Cache 读访问次数计数。
0xC	数据 Cache 读访问次数计数。
0xD	二级 Cache 访问次数计数（仅指指令流和数据流请求读 STAG 的次数）。
0xE	主存访问次数计数（只包含指令流和数据流读主存的次数）。
0xF	Cache 一致性请求次数计数（只包含多 Cache 一致性处理产生的请求）。

事件计数器 1 的计数事件由 CSR: PC1_CR[PC1_SEL(4:0)]决定，具体定义如表 8-3。

表 8-3: 事件计数器 1 的计数事件

PC1_SEL[5:0]	事件计数器 1 计数事件
0x0	指令流水线空周期计数（流水线空含义是指令译码、寄存器重命名、发射和退出站台都为空）。
0x1	重命名站台因整数重命名寄存器不足引起指令流水线停顿周期计数。

0x2	重命名站台因浮点重命名寄存器不足引起指令流水线停顿周期计数。
0x3	重命名站台因整数等待队列 (IWQ) 满引起指令流水线停顿周期计数。
0x4	重命名站台因浮点等待队列 (FWQ) 满引起指令流水线停顿周期计数。
0x5	重命名站台因重排序缓冲 (ROB) 满引起指令流水线停顿周期计数。
0x6	发射站台因装入队列 (LQ) 满引起指令发射停顿周期计数。
0x7	发射站台因存储队列 (SQ) 满引起指令发射停顿周期计数。
0x8	发射站台空的周期计数。
0x9	整数指令发射队列 IQ0 有阻塞情况的周期计数。
0xA	ROB 头部为访存类指令, 且无法退出的周期计数。
0xB	执行站台判断出所有转移指令预测失败的次数。
0xC	执行站台判断出条件转移指令预测失败的次数。
0xD	执行站台判断出无条件转移指令预测失败的次数。
0xE	执行站台判断出 CALL、JMP 指令预测失败的次数。
0xF	执行站台判断出 RET 指令预测失败的次数。
0x10	二级 Cache 脱靶次数计数, 仅指指令流和数据流读 STAG 发生 Miss 的次数。
0x11	二级 Cache 管理部件记录访存请求直接命中存储器页次数计数, 根据存储控制器的响应 ReadDataDirty 和 ReadData 携带的信息计数。
0x12	二级 Cache 管理部件记录访存请求命中存储器页次数计数, 根据存储控制器的响应 ReadDataDirty 和 ReadData 携带的信息计数。
0x13	二级 Cache 管理部件记录访存请求不命中存储器页次数计数, 根据存储控制器的响应 ReadDataDirty 和 ReadData 携带的信息计数。
0x14	Cache 一致性请求命中二级 Cache 次数(包括命中回写缓冲), 仅指多 Cache 一致性处理产生的一致性请求。
0x15	BPU 误预测次数, 即二级预测修复一级预测的次数。
0x16	指令 Cache 读访问脱靶次数计数 (即不命中 ICache, 也不命中 ICache 装填缓冲)。
0x17	ITB 脱靶次数计数。
0x18	L1DTB Miss, L2DTB 命中或 L2DTB 小页偶校验错的计数
0x19	保留
0x1A	同站台旁路冲突计数
0x1B	MAF 满冲突计数
0x1C	同站台索引冲突计数
0x1D	DCache 端口 0 冲突计数

0x1E	DCache 端口 1 冲突计数
0x1F	写寄存器端口冲突计数
0x25	访存请求从其它核心的 Cache 读取数据的次数计数。
0x26	整数指令发射队列 IQ1 有阻塞情况的周期计数。
0x27	整数指令发射队列 IQ2 有阻塞情况的周期计数。
0x28	访存指令地址发射队列 AQ0 有阻塞情况的周期计数。
0x29	访存指令地址发射队列 AQ1 有阻塞情况的周期计数。
0x2a	整数存储指令数据发射队列 ISQ 有阻塞情况的周期计数。
0x2b	浮点存储指令数据指令发射队列 FSQ 有阻塞情况的周期计数。
0x2c	浮点指令发射队列 FQ0 有阻塞情况的周期计数。
0x2d	浮点指令发射队列 FQ1 有阻塞情况的周期计数。
0x2e	重命名站台因访存等待队列 (AWQ) 满引起指令流水线停顿周期计数。
0x30	DTB SingleMiss 计数
0x31	DTB Double Miss 计数
0x32	DCache Miss 计数
0x33	Load-Load 乱序自陷计数
0x34	Load_Store 乱序自陷计数
0x35	保留
0x36	总的自陷计数
0x37	不含预取请求的访存指令发生 CacheMiss 的计数
其它	保留。

选择处理器周期计数时，每个时钟周期计数器都“+1”；选择其它周期计数时，指定的计数条件成立时，每个时钟周期计数器“+1”。选择次数计数时，指定的事件每发生一次，计数器“+1”，需注意有些事件在一个周期里可能发生多次。

8.3 事件计数的流程

事件计数的一般流程如下：

1) 配置相关的 CSR，先通过写 CSR: PC0_CR[PC0_SEL]与 CSR: PC1_CR[PC1_SEL]选择事件计数事件，写 CSR: PC0_CR[PC0]与 CSR: PC1_CR[PC0]设置事件计数器的初值，然后写 CSR: IS_CTL[PC0_EN]或 CSR: IS_CTL[PC1_EN]指定事件计数中断 0 或 1 使能，再写 CSR: IS_CTL[CM_PCE]和 CSR: IS_CTL[PC_CM]，或只写 CSR: IS_CTL[AM_PCE]指定事件计数事件所在模式，最后写 CSR: IER0~3[PCEN]，打开事件计数中断使能(注意：如果希望准确计数，则需要先将 CSR: IS_CTL[PC0_EN]或 CSR: IS_CTL[PC1_EN]关闭，配置好 CSR: C0_CR 和 CSR:PC1_CR

后再将计数使能打开);

2) 计数器计数: 事件计数使能打开后, 硬件就会根据相应 CSR 选择的计数事件, 每发生一次对应事件, CSR: PC0_CR[PC0]或 CSR: PC1_CR[PC1] “+1”;

3) 计数器溢出: 当 PC0_CR 或 PC1_CR 中的计数器溢出时, 若事件计数中断使能, 则产生事件计数中断请求, 硬件置 INT_STAT0~3 中 PC[1:0]对应位有效;

4) 硬件中断: 核心响应事件计数中断, 进入中断特权处理程序;

5) 中断处理: 事件计数器溢出后从 0 开始重新计数, 中断服务程序需要重新设置事件计数器的初值, 为以后的计数作准备。

一些常用的事件计数内容和对应的事件计数事件的选择方法详见附录 B。

9 功耗管理

申威 411 处理器提供的功耗管理机制，支持多种低功耗运行模式，软件可以根据当前程序的运行情况，执行某些特殊操作，协助硬件进入不同层次的低功耗运行模式。

9.1 核心深睡眠

当申威 411 处理器中某个核心没有可执行的任务时，可向该核心发送睡眠中断请求，使它进入深睡眠状态。睡眠的核心将不再运行任何程序，处于睡眠复位状态，并大幅度降低工作时钟频率，从而大大降低其运行功耗。申威 411 处理器允许 16 个核心全部进入睡眠状态。当需要申威 411 处理器核心恢复运行时，处于非睡眠状态的核心可向睡眠核心发送睡眠唤醒中断请求，或者外部系统通过维护接口或以太网接口向申威 411 处理器发送唤醒请求，来启动指定核心或所有核心恢复正常运行。

9.2 核心浅睡眠

在非 HM 模式下，核心执行停机指令（HALT），指令流水线将停止取指，此时该核心处于一种浅睡眠状态，即不执行任何指令，也不访问核心内所有 Cache 和存储器，但仍然支持 Cache 一致性操作，寄存器和 Cache 中的数据不会丢失。虽然浅睡眠状态时钟仍然运行在高频率状态，但配合硬件实现的门控时钟技术，仍然可大大降低停机期间的核心运行功耗。通过任何未屏蔽的中断可将处于浅睡眠状态的核心唤醒，恢复正常运行。

9.3 关闭浮点部件时钟

当软件判断出申威 411 处理器核心上执行的程序中不出现扩展指令时，可通过写 CSR：UPCR[SCE]为“0”，强制关闭浮点部件中从流水线的时钟，从而降低核心运行功耗。

当软件判断出申威 411 处理器核心上执行的程序中不出现扩展指令和浮点指令，可通过写 CSR：UPCR[FCE]为“0”，强制关闭整个浮点部件的时钟，从而降低核心运行功耗。

9.4 动态功耗调整

当软件判断出可以降低申威 411 处理器核心上程序执行的速度时，可通过写 CSR：ISSUE_CTL 寄存器，降低当前核心发射指令的速度，从而减少整个核心的信号翻转率，配合硬件上实现的门控

时钟技术，降低核心运行功耗。

申威 411 处理器支持核心基于指令流的调试，也支持基于数据流的调试。

10 调试支持

10.1 指令流调试支持

10.1.1 SYS_CALL 指令

SYS_CALL 指令根据不同的功能码，可以分解为多种不同功能的指令，具体 SYS_CALL 指令功能与软件特别是操作系统相关，但一般都有两个基本的非特权 SYS_CALL 指令用于指令流的调试，分别是断点自陷 SYS_CALL 和单步自陷 SYS_CALL。

断点自陷 SYS_CALL 功能是中止当前指令流处理，并将核心的控制权交给操作系统。使用时，在程序中需要设置断点的位置，用这条 SYS_CALL 指令代替原有指令，并将原有指令保存起来。程序运行时，当指令流到达断点自陷 SYS_CALL 指令时，将自陷到硬件模式(HM)，通过 CSR:EXC_PC 获得断点处的虚地址，保存此虚地址，并将此虚地址处的指令恢复为原有的指令，再转移到操作系统运行环境，进行进一步处理，从而实现指令流的断点调试。

单步自陷 SYS_CALL 类似于断点自陷 SYS_CALL 指令，不同在于单步自陷 SYS_CALL 能自动设置下一个断点，该断点位置为当前断点指令执行后，将要执行的下一条指令（不一定是程序顺序的下一条指令，因为存在转移等改变程序运行轨迹的指令），从而实现指令一条一条依次单步执行。

10.1.2 指令流地址匹配

指令流调试时，可能需要确定指令流是否会执行到某个虚地址或某段虚地址范围，此时可通过 CSR: IA_MATCH 和 CSR: IA_MASK 来匹配这个指令流地址或地址范围。CSR: IA_MATCH 作为指令流地址匹配寄存器，存放需要跟踪的处理器模式、指令流虚地址、虚拟机号和用户进程号，CSR: IA_MASK 作为指令流地址匹配的屏蔽寄存器，可以实现模糊跟踪（只匹配部分虚地址信息）。当指令流虚地址及其相关信息和受 CSR: IA_MASK 屏蔽后 CSR: IA_MATCH 的内容匹配时，产生指令流故障。在相应的特权处理程序中，如果 CSR: EXC_SUM[IA_MATCH]为“1”，则表示指令流地址匹配成功，此时 CSR: EXC_PC 存放的是匹配的指令流虚地址（也是被调试程序的断点虚地址）。

10.1.3 指令流跳转目标地址匹配

指令流调试时，可能需要确定是哪条指令引起指令流跳转到某个目标虚地址或某段虚地址范围，此时可通过 CSR: IDA_MATCH 和 CSR: IDA_MASK 来匹配这个指令流的跳转目标地址或地址范围。CSR: IDA_MATCH 作为指令流目标地址匹配寄存器，存放需要跟踪的处理器模式和指令流虚

地址，CSR: IDA_MASK 作为指令流目标地址匹配的屏蔽寄存器，可以实现模糊跟踪（只匹配部分虚地址信息）。当整数转移目标指令地址及其转移目标的处理器模式和受 CSR: IDA_MASK 屏蔽后 CSR: IDA_MATCH 的内容匹配时，产生指令流故障。在相应的特权处理程序中，如果 CSR: EXC_SUM[IDA_MATCH]为“1”，则表示指令流跳转目标地址匹配成功，此时 CSR: EXC_PC 存放的是这个跳转目标的虚地址，CSR: IVA 中存放的是跳转指令的 PC 值。

10.2 数据流调试支持

申威 411 处理器支持数据流调试，但需要特权程序和操作系统环境的支持。申威 411 处理器设置了 CSR: DA_MATCH 作为数据流地址匹配寄存器，CSR: DA_MASK 作为地址匹配的屏蔽寄存器，当数据流地址满足这两个 CSR 确定的匹配要求时，产生数据流故障，进入相应的特权程序。相关流程如下：

1) 设置 CSR: DA_MATCH，最小地址比较的粒度为字节，有效的物理地址为 PA[39:0]位，有效的虚地址为 VA[42:0]位；

2) 设置 CSR: DA_MASK，确定参与匹配比较的地址位，可以用于扩大地址比较的粒度（将屏蔽位的低位设置为“0”）或实现部分地址位的比较（将不比较的地址位对应的屏蔽位设置为“0”）；

3) 核心对访存指令的数据流地址进行匹配判断，当数据流地址和受 CSR: DA_MASK 屏蔽后 CSR: DA_MATCH 的内容匹配或不匹配时（根据 CSR: DA_MATCH[MATCH]确定），在相关 CSR 中登记相关信息，作为数据流故障，进入特权处理程序；

4) CSR: DVA 中存放匹配的数据流虚地址，CSR: EXC_PC 中存放产生该数据流虚地址的访存指令 PC 值，CSR: EXC_SUM[REG(4:0)]中存放该指令的寄存器 Ra 域，CSR: DS_STAT[OPCODE(5:0)]中存放该指令的操作码，CSR: DS_STAT[WR]指示该指令是读还是写访问；

5) 进入特权处理程序后，特权处理程序首先排除不是虚地址符号扩展错、访问越权、读访问故障和写访问故障后，再进入数据流地址匹配的处理。

通过这种方式，在操作系统的支持下，可实现对数据流地址的跟踪，既可全地址跟踪，也可部分地址跟踪；既可物理地址跟踪，也可实现虚地址跟踪。需要注意以下问题：

1) 对于 PRI_LDx/p、PRI_LDx_INC/p、PRI_LDx_DEC/p、PRI_LDx_SET/p 和 PRI_STx/p 指令产生的数据流地址直接当作物理地址，不进行 DTB 转换，因此只能进行物理地址匹配；

2) VM 模式下，数据流虚地址[42]位为“0”时，作为超页地址，即可进行物理地址匹配，也可进行虚地址匹配；

3) 除上述情况之外，将访存指令产生的数据流虚地址经过 DTB 转换后的地址作为数据流物理地址。

11 I/O 虚拟化支持

申威 411 处理器集成的 PCI-E 接口支持 IOMMU 虚拟化机制，用来对设备的 DMA 访问进行地址空间的重新映射，有两种可选择的方式：段式地址代换和页式地址代换。

11.1 段式地址代换

段式地址代换中，每个 PCI-E 接口内设置了一套 64 条目的 I/O 地址代换表，分别对应 64 个 IOR 寄存器 IOMMU0~IOMMU63，地址代换表的装填和管理完全由软件负责，可通过写 IOR 实现。

当 PCI-E 接口收到一个 DMA 请求时，先将 8 位总线号，5 位设备号和 3 位功能号拼接为 16 位的标记信息（tag 号），然后查找 64 条目的地址代换表，处理流程如下：

1) 如果命中其中一个条目，且地址在允许访问的空间范围内，则将 DMA 请求携带的地址加上条目中的基址，形成真正的访存地址，发向指定的存储控制器；

2) 如果命中其中一个条目，但地址超过了允许访问的空间范围，则产生 DMA 请求地址越界异常，在 IOR: IOMMU 地址代换异常寄存器 (IOMMUException) 中记录出错的条目和出错的类型（越界），转 4)；

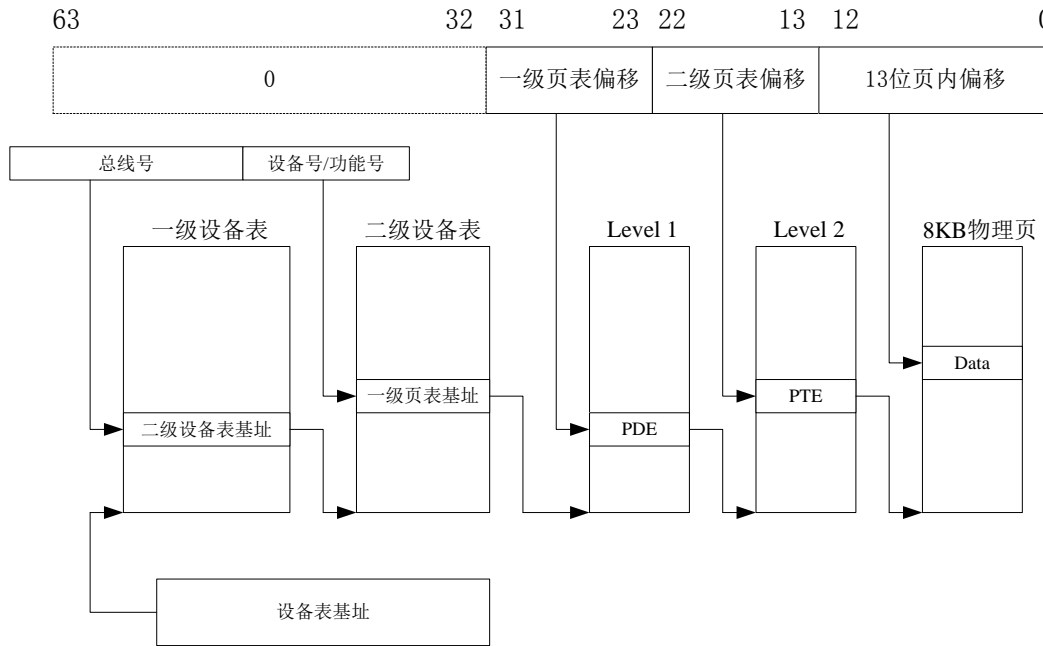
3) 如果不命中所有条目，则产生 DMA 请求地址缺失异常，在 IOR: IOMMU 地址代换异常寄存器中记录出错的 DMA 请求的 tag 号和出错的类型（缺失），转 4)；

4) 产生 MSI 或 INTA/B/C/D 中断发给某个核心，具体的中断类型和处理中断的核心号，软件可通过写 IOR: IOMMU 地址代换异常寄存器中的某些字段来指定；核心收到异常中断后进行相应的处理。

注意：发生越界、缺失异常的 DMA 请求会被硬件直接丢弃掉。

11.2 页式地址代换

页式地址代换中设置了 4 级表格：一级设备表、二级设备表、一级页表、二级页表。其代换的整体过程如图 11-1 所示。


图 11-1 PCI-E 页式地址代换的 4 级空间管理方式

代换过程为：

- 1) 使用“设备表基地址”（Device Table Base Address）加上地址总线号对应的偏移（ $\times 8$ ）形成地址，访问“一级设备表”，得到“二级设备表基地址”；
- 2) 使用“二级设备表基地址”加上地址设备号、功能号对应的偏移（ $\times 8$ ）形成地址，访问“二级设备表”，得到“一级页表基地址”；
- 3) 使用“一级页表基地址”加上地址[31:23]对应的偏移（ $\times 8$ ）形成地址，访问“一级页表”，得到“二级页表基地址”；
- 4) 使用“二级页表基地址”加上地址[22:13]对应的偏移（ $\times 8$ ）形成地址，访问“二级页表”，得到“物理页表项”（PTE）；
- 5) 最终的访问地址就是物理页表项内容作为基地址，加上地址[12:0]作为页内偏移。

11.2.1 设备表

设备表用于存放各设备的一级页表基址，由操作系统维护。为了兼顾 IOMMU 快速访问设备表以及节省设备表对系统主存空间的占用，将设备表实现为两级设备表结构。两级设备表均为线性页表，其中一级设备表总共 256 个条目，由总线号索引，二级设备表分为 256 组（实际数量根据系统中 PCIE 总线的数量确定，在主存中可以离散放置），每组 256 个条目，组内由设备号+功能号索引。

- 1) 一级设备表

一级设备表由总线号索引，每个条目存放该总线对应的二级设备表的基地址（系统主存地址），占用 64 位的主存空间，每个条目包含：

- 有效位（1 位）；
- 二级设备表的基地址（物理地址[39:13]）；

2) 二级设备表

二级设备表由地址中的设备号+功能号索引，每个条目存放该 PCI 设备对应的一级页表基地址（系统主存地址），占用 64 位的主存空间，每个条目包含：

- 有效位（1 位）；
- 一级 IO 页表的基地址（物理地址[39:13]）；
- DMA 地址转换使能位（如果不使能，该设备的 DVA 就是物理地址，直接访存。直接使用物理地址访问时没有读写权限检查的限制）。

11.2.2 IO 页表

页表用于各设备 DMA 操作的虚地址（DVA）到主存物理地址的代换，由操作系统维护，采用两级表结构。页表的实际数量由操作系统根据运行过程中 DMA 操作实际需要的主存空间大小，进行动态分配和释放。

1) 一级页表：

一级页表根据一级页表基址并由 DVA[31:23]索引，每个条目存放该设备（BusID、DeviceID、FuncID）虚地址对应的二级页表基地址（系统主存地址），占用 64 位的主存空间，条目内容如下：

- 有效位（1 位）；
- 二级页表的基地址（物理地址[39:13]）；

2) 二级页表：

二级页表根据二级页表基址并由 DVA[22:13]索引，每个条目存放该设备虚地址对应物理地址（系统主存地址），占用 64 位的主存空间，条目内容如下：

- 有效位（1 位）；
- 物理页帧号（物理地址[39:13]）；
- 页面粒度（2 位，支持四种粒度）；
- 权限位（2 位读、写使能。只有在使能时才允许对该空间进行 DMA 读、写）。

附录 A CSR 的限制

虽然在申威 411 处理器硬件上通过 CSR 记分板和 CSR 写机制，保证大多数情况下 CSR 的隐式读写和显式读写之间的相互顺序，但还有些特殊情况，需要软件协助保证。另外，还有些 CSR 在硬件实现时存在一定关联，因此软件显式写这些 CSR 时有一定的限制。

1) 指令部件的 CSR 显式写和隐式读顺序

指令部件中 CSR 的隐式读产生在指令发射之前，这些 CSR 通过硬件上的 CSR 记分板机制无法保证显式写与隐式读之间的顺序，为确保这些 CSR 的修改对后续指令产生预期的效果，需要进行如下操作。

a. 对于 CSR: IS_CTL[F_BAD_DPAR]、IS_CTL[F_BAD_TPAR]和 IC_FLUSH，在 HM 模式或非 HM 模式下取指就存在隐式读，因此需特别小心，可将 PRI_WCSR 指令、读 PRI_CSR 指令（读指令部件的 CSR）和 PRI_RET/b 指令放在一个指令组中（对界的 4 条指令），以保证再次取指令时，这些 CSR 的值已经被更新；

b. 对于 CSR: IS_CTL[SRE]，在 HM 模式下存在隐式读，需要在 PRI_WCSR 指令之后，在使用系统寄存器的指令之前，插入一条 IMB 指令；

c. 对于 CSR: UPCR[UPN]、VPCR、ITB_TAG、ITB_PTE、ITB_IA、ITB_IV、ITB_IVP、ITB_IU、ITB_IS，只在非 HM 模式下取指才会存在隐式读，可在 PRI_WCSR 指令后直接用 PRI_RET/b 返回；

d. 对于 CSR: IS_CTL[16:10]、IA_MATCH、IA_MASK，如果要用于 HM 模式，则可采用类似 b 的处理方法，如果用于非 HM 模式，则可在 PRI_WCSR 指令和 PRI_RET/b 之间，插入一条 PRI_RCSR 指令（读指令部件的 CSR）；

e. 对于 CSR: INT_CLR0~3、IER0~3、II_CLR、II_ER，如果在 HM 模式下进行了显式写操作，可在 PRI_WCSR 指令和 PRI_RET/b 之间，插入一条 PRI_RCSR 指令（读指令部件的 CSR），避免已处理过的中断再次被处理或已屏蔽的中断被处理。

2) 数据 Cache 管理部件和二级 Cache 管理部件的 CSR 显式写和隐式读顺序

数据 Cache 管理部件和二级 Cache 管理部件的 CSR 隐式读主要由访存指令引起，硬件上已经保证了这些 CSR 的显式写和隐式读之间的顺序问题。

3) 整数部件的 CSR 显式写和隐式读顺序

a. 对于大多数访存指令，只对数据 Cache 管理部件的 CSR 进行隐式读，但对原子操作指令而言，还需要对整数部件的 CSR: ICR 进行隐式读，而硬件并没有对此进行控制，因此软件需要

在写CSR: ICR之后和原子操作指令之前，增加一条读整数部件CSR的指令；

b. RTC指令，需要隐式读CSR: TC，RTC指令读访问时受CSR记分板控制，但由于TC每个周期都在隐式写，而硬件可能出现乱序执行RTC指令的情况，因此RTC指令返回的周期计数可能不准确，如果软件希望获得某条指令执行后准确的周期数，可以使得RTC指令中Rb域内容与前一条指令存在寄存器写后读相关性；

c. 对于CSR: CID，既可通过PRI_RCRS指令显式读，也可通过RCID指令显式读，PRI_RCSR指令和RCID指令读访问时都受CSR记分板控制。另外，CSR: CID中的值不存在硬件隐式写和隐式读操作，因此不存在显式读写与隐式读写之间的顺序问题；

d. 在进行指令流整数转移指令目标地址匹配时，注意写IDA_MATCH和IDA_MASK的指令地址与真正要匹配的目标指令地址必须有一定的距离（避免PC值接近而引起错误操作）。

4) CSR 显式写和隐式写顺序

一般情况下，CSR不存在同时显式写和隐式写，只有那些可写“1”清除或写清除的CSR会出现这种情况，如CSR: DC_STAT、DS_STAT、SC_STAT等，这时需要软件保证正确性。一般可以在显式写这些CSR指令后，插入MEMB指令来避免同时产生显式写和隐式写的情况。

5) 异常相关的 CSR

因异常自陷进入特权程序后，必须在特权程序的开始，立即读相关状态的CSR，否则因特权程序的运行，会影响到CSR的内容。这类CSR包括IVA、IVA_FORM、INT_STAT0~3、EXC_SUM、DS_STAT、DVA和DVA_FORM等。

在异常或自陷特权程序中，对已处理的异常或自陷状态需要予以清除，否则这些状态会保持，影响后继的异常或自陷处理。这类CSR包括IS_STAT、DC_STAT、DS_STAT、SC_STAT等。

6) 显式写受限制的 CSR

根据申威411处理器的硬件实现，软件显式写某些CSR时需受一定限制，如：为简化指令Cache内容，指令Cache中不包含虚拟机号VPN，因此当软件更改CSR: VPCR[VPN]时，必须同时写CSR: IC_FLUSH，刷新指令Cache。

附录 B 事件计数内容与方法

申威 411 处理器设置了两个事件计数器（PC0 和 PC1），软件可根据需要，让 PC0 和 PC1 选择不同的事件同时计数，一个计数器的值作为分母，一个计数器的值作为分子，可计算出比值后用于衡量申威 411 处理器的相关性能。应尽可能使得需要的事件计数值在申威 411 处理器上运行一次得到，以提高事件计数的准确性。下面列举了一些常用的事件计数内容及其计数事件的配置方法。

1) 转移预测事件计数的配置方法如表 B-1 所示。

表 B-1: 转移预测事件计数的配置方法

性能测试事件	事件计数器 0	事件计数器 1
总转移预测失败率	PC0_SEL=0x3 执行站台执行的转移指令计数	PC1_SEL=0xB 执行站台判断出所有转移指令预测失败的次数
条件转移指令的转移预测失败率	PC0_SEL=0x4 执行站台执行的条件转移指令计数	PC1_SEL=0xC 执行站台判断出条件转移指令预测失败的次数
无条件转移指令的转移预测失败率	PC0_SEL=0x5 执行站台执行的无条件转移指令计数	PC1_SEL=0xD 执行站台判断出无条件转移指令预测失败的次数
JMP、CALL 的转移预测失败率	PC0_SEL=0x6 执行站台执行的 JMP、CALL 指令计数	PC1_SEL=0xE 执行站台判断出 JMP、CALL 指令预测失败的次数
RET 的转移预测失败率	PC0_SEL=0x7 执行站台执行的 RET 指令计数	PC1_SEL=0xF 执行站台判断出 RET 指令预测失败的次数
条件转移指令占转移指令比例	PC0_SEL=0x3 执行站台执行的转移指令计数	PC0_SEL=0x4 执行站台执行的条件转移指令计数
无条件转移指令占转移指令比例	PC0_SEL=0x3 执行站台执行的转移指令计数	PC0_SEL=0x5 执行站台执行的无条件转移指令计数
JMP、CALL 指令占转移指令比例	PC0_SEL=0x3 执行站台执行的转移指令计数	PC0_SEL=0x6 执行站台执行的 CALL 和 JSR 指令计数
RET 指令占转移指令比例	PC0_SEL=0x3 执行站台执行的转移指令计数	PC0_SEL=0x7 执行站台执行的 RET 指令计数
转移预测的误预测率	PC0_SEL=0x3 执行站台执行的转移指令计数	PC1_SEL=0x15 预测的指令类型与实际的指令类型不同的次数。

2) 指令流水线事件计数的配置方法如表 B-2 所示。

表 B-2: 指令流水线事件计数的配置方法

性能测试事件	事件计数器 0	事件计数器 1
CPU 空闲概率	PC0_SEL=0x8 周期计数	PC1_SEL=0x0 指令流水线空闲周期计数
整数寄存器不足引起的指令流水线阻塞的比例	PC0_SEL=0x8 周期计数	PC1_SEL=0x1 重命名站台因整数重命名寄存器不足引起指令流水线停顿周期计数
浮点寄存器不足引起的指令流水线阻塞的比例	PC0_SEL=0x8 周期计数	PC1_SEL=0x2 重命名站台因浮点重命名寄存器不足引起指令流水线停顿周期计数
IQ 容量不足引起的指令流水线阻塞的比例	PC0_SEL=0x8 周期计数	PC1_SEL=0x3 重命名站台因整数发射队列 (IQ) 满引起指令流水线停顿周期计数
FQ 容量不足引起的指令流水线阻塞的比例	PC0_SEL=0x8 周期计数	PC1_SEL=0x4 重命名站台因浮点发射队列 (FQ) 满引起指令流水线停顿周期计数
ROB 容量不足引起的指令流水线阻塞的比例	PC0_SEL=0x8 周期计数	PC1_SEL=0x5 重命名站台因重排序缓冲 (ROB) 满引起指令流水线停顿周期计数
LQ 容量不足引起的指令发射阻塞的比例	PC0_SEL=0x8 周期计数	PC1_SEL=0x6 发射站台因装入队列 (LQ) 满引起指令流水线停顿的周期计数
SQ 容量不足引起的指令发射阻塞的比例	PC0_SEL=0x8 周期计数	PC1_SEL=0x7 发射站台因存储队列 (SQ) 满的周期计数停顿的周期计数
取指和译码速度不足引起指令流水线断流的比例	PC0_SEL=0x8 周期计数	PC1_SEL=0x8 发射站台因缺少指令而无法发射的周期计数
指令发射站台因各种原因无法全速发射的比例	PC0_SEL=0x8 周期计数	PC1_SEL=0x9 发射站台阻塞发射的周期计数

3) 访存流水线事件计数的配置方法如表 B-3 所示。

表 B-3: 监测访存流水线性能的配置方法

性能测试事件	事件计数器 0	事件计数器 1
ITB 脱靶率	PC0_SEL=0x9 ITB 访问次数计数	PC1_SEL=0x17 ITB 脱靶次数计数
DTB Single 脱靶率	PC0_SEL=0xA DTB 访问次数计数	PC1_SEL=0x30 DTB SingleMiss 次数计数
DTB Double 脱靶率	PC0_SEL=0xA DTB 访问次数计数	PC1_SEL=0x31 DTB Double Miss 次数计数
指令 Cache 脱靶率	PC0_SEL=0xB	PC1_SEL=0x16

	指令 Cache 访问次数计数	指令 Cache 脱靶次数计数
数据 Cache 脱靶率	PC0_SEL=0xC 数据 Cache 访问次数计数	PC1_SEL=0x32 数据 Cache 脱靶次数计数
SCache 脱靶率	PC0_SEL=0xD 主存访问次数计数	PC1_SEL=0x10 SCache 脱靶次数计数
主存直接页命中率	PC0_SEL=0xE 主存访问次数计数	PC1_SEL=0x11 访存请求直接命中存储器页次数计数
主存页命中率	PC0_SEL=0xE 主存访问次数计数	PC1_SEL=0x12 访存请求命中存储器页次数计数
主存页脱靶率	PC0_SEL=0xE 主存访问次数计数	PC1_SEL=0x13 访存请求不命中存储器页次数计数
访存请求从其它核心的 Cache 读取数据的概率	PC0_SEL=0xE 主存访问次数计数	PC1_SEL=0x25 访存请求从其它核心的 Cache 读取数据的次数计数。
核外一致性机制对副本记录的精确率	PC0_SEL=0xF Probe 请求次数计数	PC1_SEL=0x14 Probe 请求查询命中 SCache 次数
读读重发自陷发生比例	PC0_SEL=0x1 退出的访存指令计数	PC1_SEL=0x33 读-读重发自陷次数计数
读写重发自陷发生比例	PC0_SEL=0x1 退出的访存指令计数	PC1_SEL=0x34 写-读重发自陷次数计数
DBOX 重发自陷发生比例	PC0_SEL=0x1 退出的访存指令计数	PC1_SEL=0x36 数据 Cache 管理部件重发自陷总次数计数
非预取指令发生 DCacheMiss 的次数	PC0_SEL=0xC 数据 Cache 访问次数计数	PC1_SEL=0x37 非预取指令发生 DCacheMiss 的次数
L1DTBmiss, L2 DTB 命中或偶校验导致的自陷发生比例	PC0_SEL=0x1 退出的访存指令计数	PC1_SEL=0x18 L1 DTB 命中, L2DTBmiss 或 L2DTB 小页偶校验错导致的自陷
旁路 SQ 数据未到达冲突发生比例	PC0_SEL=0x1 退出的访存指令计数	PC1_SEL=0x19 Store 数据未到达, 导致 Load 指令冲突次数计数
同站台冲突发生比例	PC0_SEL=0x1 退出的访存指令计数	PC1_SEL=0x1a 流水线 A、B 之间产生的冲突
MAF 满冲突发生比例	PC0_SEL=0x1 退出的访存指令计数	PC1_SEL=0x1b MAF 满导致的冲突
同索引同路冲突发生比例	PC0_SEL=0x1 退出的访存指令计数	PC1_SEL=0x1c MAF 或 SQ 判处的通路同索引冲突
DCache 端口 0 冲突发生比例	PC0_SEL=0x1 退出的访存指令计数	PC1_SEL=0x1d 流水线 A 与 C 发生的 DCache 端口 0 冲突
DCache 端口 1 冲突发生比例	PC0_SEL=0x1 退出的访存指令计数	PC1_SEL=0x1e 流水线 B 与 D 发生的 DCache 端口 1 冲突

流水线 B 与捎带响应 写寄存器冲突发生比 例	PC0_SEL=0x1 退出的访存指令计数	PC1_SEL=0x1f 流水线 B 与捎带响应发生的写寄存器冲 突
-------------------------------	--------------------------	--

附录 C CHIP_ID 定义

在 CSR: IS_CTL 中, CHIP_ID 域指示申威 411 处理器的 ID 号, 这个域共有 6 位, 具体定义如下:

1) [5:4]位: 区分不同结构的芯片, 其中“01₂”指示 SW-2 系列, “10₂”: 指示是 SW-3 系列, “11₂”: 指示是 SW-4 系列;

2) [3:2]位: 区分不同的工艺实现, 为“00₂”指示 65nm 工艺的实现, 为“01₂”指示 130nm 工艺的实现; 为“10₂”指示 45nm 工艺的实现;;

3) [1:0]位: 区分不同的流片, 为“00₂”指示第一次流片, 为“01₂”指示第二次流片, 如此类推。

附录 D 各种中断的初始向量值

上电复位或冷复位结束后，CSR：INT_STAT0~3 中的各种中断对应的向量位置如表 D-1 所示。

表D-1: INT_STAT3寄存器复位后对应的中断

名称	范围	类型	描述
SLEEP	[63]	RO	睡眠中断请求。
MCHK_INT	[62]	RO	机器检查错中断请求。
CR	[61]	RO	已纠正错中断请求。
PC1	[60]	RO	事件计数1中断请求。
PC0	[59]	RO	事件计数0中断请求。
TIMER_INT	[58]	RO	定时器中断请求。
—	[57:0]	—	保留

附录 E 对软件的特殊要求

由于申威 411 处理器在 4A 芯片具体实现中的特殊情况，需要软件给予特殊的流程处理，记录如下：

1、DTag 偶校验错

数据流请求访问 DTag 出现偶校验错时，多数时候会在 CSR:DS_STAT[TAG_PERR]中记录错误标志，并进入 DFAULT 异常处理入口。在一些特殊条件下，没有在 CSR:DS_STAT[TAG_PERR]中记录错误标志，但依然会进入 DFAULT 异常处理入口。因此当硬件进入 DFAULT 异常处理入口时，软件可先按常规方法查询所有进入该入口的错误标志，如果全都无效，则说明发生了 DTag 偶校验错，此时需要对 SCache 进行全刷新，以此来纠正此错误。

2、EVICTDL 和 EVICTDG 指令错误使能

Cache 淘汰指令（EVICTDL 和 EVICTDG）错误检查使能设置在 CSR:DC_CTL[EVICTD_ERR_EN]，由于硬件实现上的原因，当该配置为 0 时，若 Cache 淘汰指令发生错误，硬件可能挂住。因此要求软件保证该配置一直配置为 1，以规避此问题。

3、ILL_INST 的指令码记录

CSR:IGL_INST 为非法指令寄存器，只读，复位时，清 0。当指令流出现非法操作码故障、指令流地址匹配成功、浮点屏蔽故障时，该寄存器表示非法指令的 32 位信息。

但在 4A 具体实现时，当发生指令流地址匹配成功或浮点屏蔽故障时，没有将指令编码记录在 IGL_INST 中。

4、NC Store 指令（或存储器不可 Cache 空间的 Store）和后续的 Load 指令的顺序

前一条指令 后一条指令	Store {0, A}	Store {1, A}	NC_store {x, A}	Load {0, A}	Load {1, A}	NC_Load {x, A}
Store {0, A}	保序	不保序	不保序	保序	保序	保序
Store {1, A}	保序	保序	保序	保序	保序	保序
NC_store {x, A}	保序	保序	保序	保序	保序	保序
Load {0, A}	保序	不保序	不保序	保序	保序	保序
Load {1, A}	保序	保序	保序	保序	保序	保序
NC_Load {x, A}	保序	保序	保序	保序	保序	保序

针对表中黄色底部分，由于 4A 设计存在缺陷，硬件无法保序。因此在需要使用 NC Store 指令

或访问地址[38]为 1 的 Store 指令的场合，建议批量使用此类指令，并遵循如下限制：先执行一个 MEMB，然后批量执行不可 Cache 写操作，最后再执行一个 MEMB。

5、核心的性能统计

以下统计内容不准确：

- 1) L1DTB 不命中且 L2 DTB 命中，或者 L2DTB 发生端口冲突的次数；
- 2) MAF 满冲突发生次数；
- 3) 同索引同路冲突发生次数；

以下统计内容在核心外部实现，核心内部无法统计：

- 1) 主存直接页命中次数；
- 2) 主存页命中次数；
- 3) 主存页脱靶次数；
- 4) 访存请求从其它核心的 Cache 读取数据的次数。