无锡先进技术研究院
Wuxi Institute of Advanced Technology

SW3231 配套维护卡使用说明

版次：1.7

无锡先进技术研究院

年　　月　　日

修改文档历史记录：

日　期	版本	说　明	修改人
2020-06-20	1.0	SW3231 维护 web 界面使用说明	郦雅晴
2020-10-12	1.2	增加功能	郦雅晴
2021-01-07	1.3	增加功能	郦雅晴
2021/02/26	1.4	新增 pc 值说明	郦雅晴
2021/03/24	1.5	新增查错方法	郦雅晴
2021/03/31	1.5	1、新增 PVT 查询功能	姚子平
		2、新增 scan_error 命令	唐金阳
		3、新增 scan_vcpucb 命令	唐金阳
		4、新增 rdmesg 命令	唐金阳
2021/04/23	1.6	修改 scan_error 截图	唐金阳
2021/07/15	1.6	新增四路系统调试功能	姚子平
2021/10/18	1.7	1、更新 6.0.0.d 版本 srom 对应的 pc 值	荣乾晶
		2、更新 DLI 查错方法	周玲静
2021/11/2	1.7	更新 BIOS rrk 打印起始地址说明	郦雅晴

目 录

软件用户手册

1 概述

1.1 标识

本文档是对无锡先进技术研究院研发的 SW3231 服务器配套维护卡系统和软件使用方法进行说明。当前版本为 V1.6。

1.2 系统概述

SW3231 配套维护卡是基于以 AST2400 为核心的基板管理控制系统，用于服务器开发人员、测试人员和服务器管理人员对 SW3231 为核心的服务器（包含单路和双路）进行日常的调试、运营和维护。该维护系统不依赖于服务器的 CPU、BIOS 或者操作系统工作，是一个独立于服务器系统独立运行的维护系统，因此具有高安全和高可靠的特性。该维护系统由无锡先进技术研究院提供开发环境和开发经费，基于上一代 SW-CPU 配套维护系统，根据新一代 SW3231 服务器的需求进行订制研发，于 2020 年 6 月完成初步版本，经过无锡先进技术研究院-基础平台研发部多人修订，至 2021 年 3 月份完成较为完善的服务器配套维护系统。该维护系统当前在 SW3231 服务器系统中运行良好，暂不兼容其他服务器平台。

1.3 文档概述

本文档提供了 SW3231 配套维护卡的系统和软件的使用方法说明，主要包含 FLASH 文件加载、确认 SW3231-CPU 的配置引脚、查看 SW3231-CPU 的配置信息以及对 SW3231-CPU 状态进行查看等维护功能。

2 引用文档

无。

3 维护系统操作入门

3.1 维护系统的首次用户

3.1.1 FLASH 文件加载

通过编程器把 FLASH_V1.bin 文件烧写到 1 片 FLASH 芯片中（8MB 大小），芯片放到 CPU0 和 CPU1 的 FLASH 座。

3.1.2 确认 CPU 配置管脚

表：4-1 CPU 的配置引脚

	CPU0	CPU1
CPUID	3b’000	3b’001
SYSTEM_SCALE（双路）	2b’01	2b’01
CFG_SEL（自引导）	2b’10	2b’10
LONGTIMECNTEN_L	接 CPU0 的 GPIO_H0	接 CPU0 的 GPIO_H0（确认是 CPU0）
LF_SEL	1b’1	1b’1
SEL_TAP	5b’11111	5b’11111
PE_ATE_MODE_H	1b’0	1b’0
TERM_ENA_H	1b’1	1b’1

3.1.3 登录小卡

将维护卡接上电源适配器，小卡上的 JTAG 和主板 CPU 的 JTAG 相连（注意区分接口是 3.3V 还是 1.8V，如图 4.1 所示），小卡插上网线，上电，等待小卡启动。

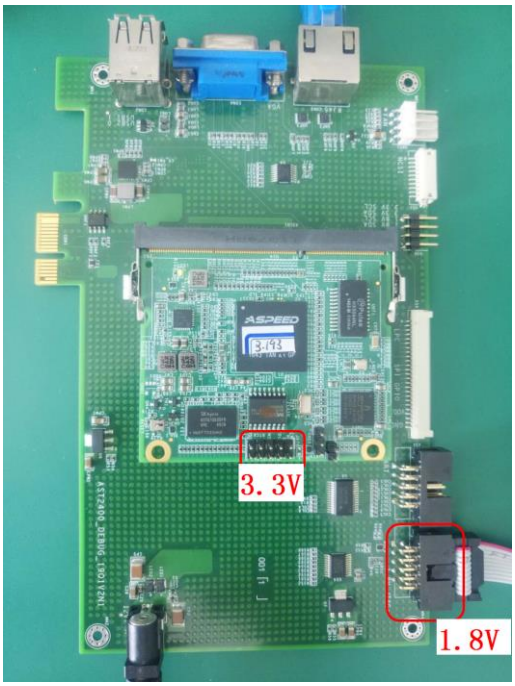


图 4.1 维护卡

如图 4.2 进入维护系统登陆界面，维护卡 IP 统一设定为 192.168.1.10 使用火狐浏览器，用户名和密码均为 user，登陆小卡。

图 4.2 维护系统登陆界面

3. 2 确认维护通路

进入维护界面,如图 4.3

图 4.3 维护系统维护界面

如果 JTAG 连接是串推，按如图 4.4 方式确认链路上的 CPU 是否能访问到，在小卡维护界面输入 mttool －rcodeid 0，mttool --rcodeid 1，mttool --rcodeid 2，mttool --rcodeid 3。如果返回均为 0x6b001，则确认 JTAG 通路无误。

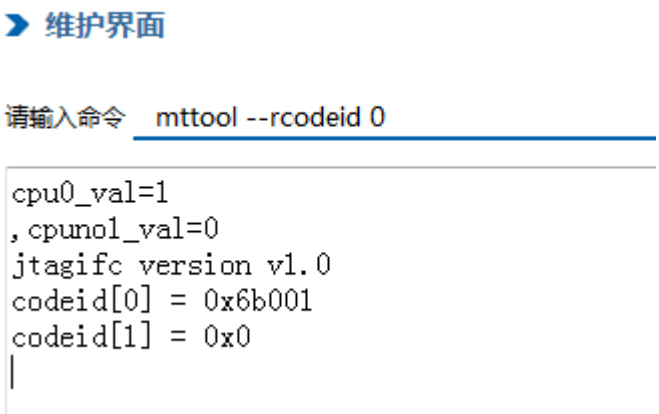


图 4.4 维护界面指令

如果 JTAG 没有串推，直接从单个 CPU 接到座，小卡每次上电都需要如下操作:

a) gpiotool --set-data-high 86

b) 然后将 JTAG 线分别接到每个 CPU，在维护界面输入命令

c) mttool --rcodeid 0，看返回是否为 0x6b001

单个 CPU 启动，如果要查看 CPU 状态，则需要将 CPU 的 SYSTEM_SCALE 和 CPUID 均接成 00。建议按照双路版本启动。

本条应概述对用户可见软件的访问与保密性方面特征。（若适用）本节应包括以下内容:

a) 如何获得与何处获得口令；

b) 如何在用户的控制下添加、删除或更改口令；

c) 与用户生成的输出报告及其它媒体的存储和加标记有关的保密性考虑。

3.3 停止和挂起

若要停止对小卡的操作，请先断开电源在对小卡进行操作。

3.4 兼容性

四路版本配套维护卡向下兼容单路、双路模式。维护卡上电默认模式为双路模式。模式切换方式如下：

单路：

gpiotool --set-data-high 86

双路：

gpiotool --set-gpios-low 86 87

四路：

gpiotool --set-gpios-high 86 87

4 使用指南

4.1 查看 CPU 配置信息

进入配置界面，如果是刚加电状态，等待 3 分钟，再操作

图 5.1 查看 CPU 配置信息

4.1.1 选择 CPUID 号

选择 CPU0、CPU1、CPU2、CPU3 进行相关信息配置

图 5.2 选择 CPUID

4.1.2 查看/修改 CPU 频率

先按读取按钮，可以看到当前 FLASH 中配置的 CPU 频率。然后按照需求设置好频率，点击烧写按钮，等待烧写成功即可。

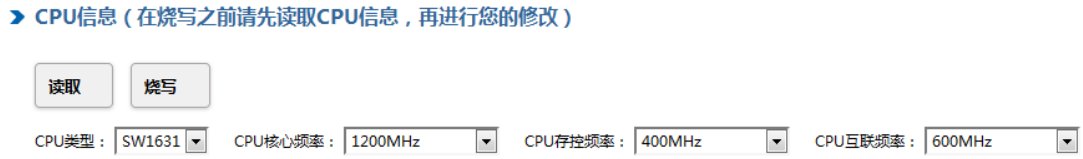


图 5.3 CPU 频率

4.1.3 查看 DLI 信息

按读取按钮，读取 DLI 当前速率，该功能只能读取 DLI 速率，不能烧写。

图 5.4 DLI 信息

4.1.4 查看 PVT 信息

按读取按钮，读取当前 CPU 的 PVT 信息，信息包括温度和电压。每个 CPU 有 3 个 PVT。

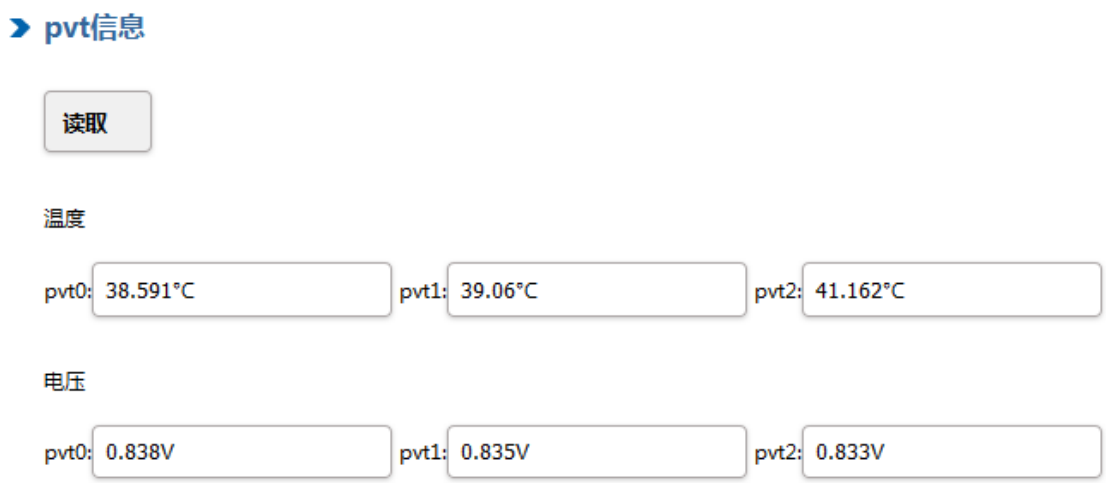


图 5.5 PVT 信息

4.1.5 查看/修改 config 信息

1） 查看/修改 core_online：先按读取按钮，可以看到当前 FLASH 中配置的 core_online 的值，core_online 重设时点击重置按钮。输入设置好的值，点击烧写，跳出烧写成功即可。

2） 查看/修改 mc_online：先按读取按钮，可以看到当前 FLASH 中配置的 mc_online 的值，点击下拉按钮，选择需要设置的 mc_online 值，点击烧写，跳出烧写成功即可。

3） 查看/修改 dli_online：先按读取按钮，可以看到当前 FLASH 中配置的 dli_online 的值，点击下拉按钮，选择需要设置的 dli_online 值，点击烧写，跳出烧写成功即可。

图 5.6 查看/修改 config 信息

4.1.6 DPC 信息配置

根据主板中每个 channel 插的是 1dimm 还是 2dimm，选择 DPC 配置，然后点击烧写，跳出烧写成功即可。

图 5.7 DPC 信息配置

4.1.7 PCIE 信息配置

按读取按钮可查看 flash 中 PCIE 拆分信息，按实际需求配置拆分信息，然后点击烧写，跳出烧写成功即可。上述信息确认烧写完成后，重启机器。

图 5.8 PCIE 信息配置

4.2 维护功能查看运行状

维护功能在左边列表维护界面中实现，如图 5.9

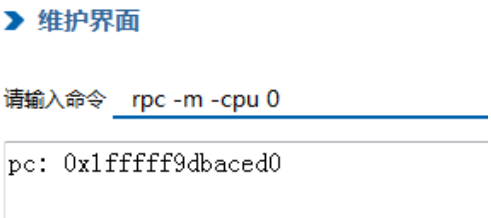
图 5.9 维护界面

4.2.1 rpc/ rpc_all

rpc <-m mpeid> [-cpu cpu_number]

-m mpeid: mpe id.

-cpu cupid，cpuid=0，选 cpu0，cpuid=1，选 cpu1，cpuid=2，选 cpu2，cpuid=3，选 cpu3



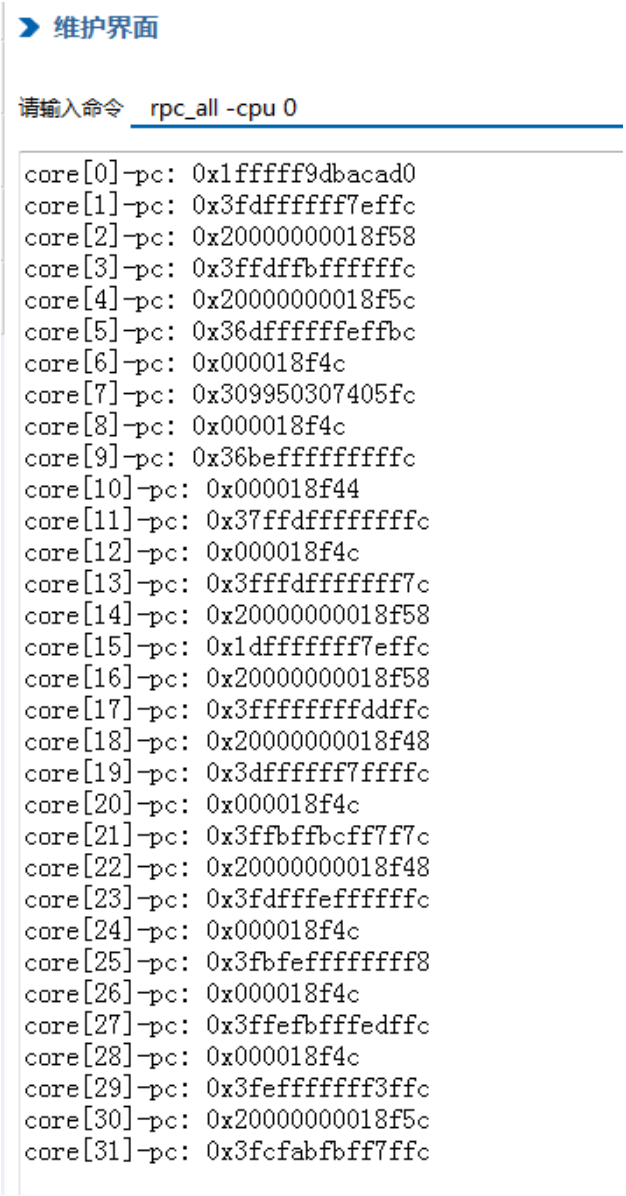


图 5.10 rpc/ rpc_all

4. 2. 2　rrk

rrk 命令打完回车后等待 15s 左右，对话框里显示出所有打印信息。

默认 rrk 打印的地址为 0x700000（内核），可通过-a 添加地址值。

对应研究院出的 bios 打印段起始地址为 0x600000

维护界面

请输入命令 rrk

```
5Linux version 4.19.90-aere-192748-ga3b25c0dca4a-dirty (wangq@platform) (SWREACH G
ID: b006536313b5fe07153d36c4576652c690d75f38 of 6B_LOCK_ERROR_AVOID) #22 SMP Wed Jul
setup_arch sys_type = 0, hwrpb = 0xfff0000000820000, hwrpb->cpuid = 0x0
bmc = 28
hwrpb->revision = 0x0.
sw_mv.vector_name = sw.
Booting on Xuelang variation Xuelang using machine vector sw
Major Options: SMP LEGACY_START DISCONTIGMEM NUMA
Command line: root=/dev/sda2 ip=172.16.137.208::172.16.137.254:255.255.255.0::eth0:o
Raw memory layout:
 memcluster  0, usage 0, start     1144, end 33554432
 memcluster  1, usage 0, start 33554432, end 67108864
Initializing bootmem allocator on Node ID 0
 memcluster  0, usage 0, start     1144, end 33554432
 Detected node memory:   start     1144, end 33554432
 freeing pages 1144:33554432
Initializing bootmem allocator on Node ID 1
 memcluster  1, usage 0, start 33554432, end 67108864
 Detected node memory:   start 33554432, end 67108864
 free1
ing pages 33554432:67108864
```

维护界面

请输入命令 rrk -a 0x810000

```
======================= BIOS INIT =======================
BMC number is 28
SW arch initialize!
cpu_online = 0x1
rc index = 0
PCIe CPU node 0 Hose-0 Link down!
rc index = 1
PCIe CPU node 0 Hose-1 Link down!
rc index = 2
PCIe: CPU node 0 Hose-2 Link up succeed!
node=0 index=2
0x100407
rc_init_busnr=41 rc_init_busnr_end=61
pci scan bus[29-00]
pci 0000:29:00.0: [8086:10d3] type 00 class 0x020000
pci 0000:29:00.0: reg 10: [mem 0xfffe0000-0x0 32bit]
pci 0000:29:00.0: reg 18: [io 0xffffffe0-0x0]
pci 0000:29:00.0: reg 1c: [mem 0xffffc000-0x0 32bit]
SW arch start to allocate pci space
pci 0000:29:00.0: BAR 0: [mem 0xe0000000-0xe0020000]
pci 0000:29:00.0: BAR 2: assigned [io 0x0-0x20]
pci 0000:29:00.0: BAR 3: [mem 0xe0020000-0xe0024000]
pci 0000:29:00.0: update reg 10: 0xe0000000 type: mem 32bit
pci 0000:29:00.0: update reg 18: 0x0 type: io 32bit
pci 0000:29:00.0: update reg 1c: 0xe0020000 type: mem 32bit
rc index = 3
```

图 5.11 rrk

4. 2. 3 dli_link_stat

在维护界面输入 dli_link_stat，查看 DLI link 状态，如图 5.12 所示.

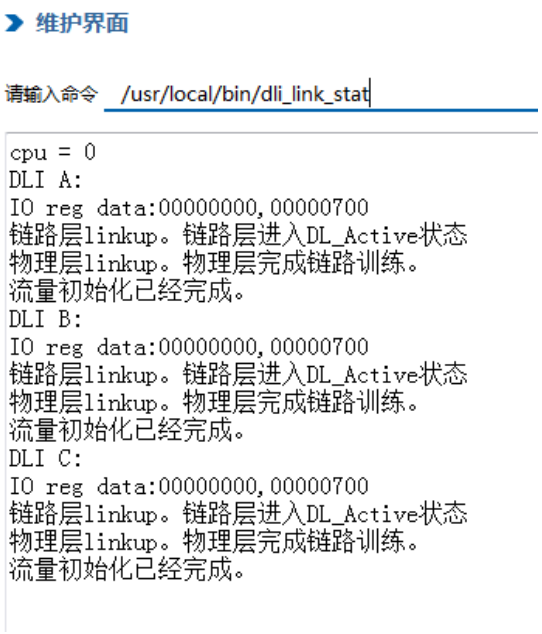


图 5.12 DLI_LINK

4. 2. 4　rd_event_cnt

查看 DLI CRC 次数，值越小说明 DLI 质量越好。

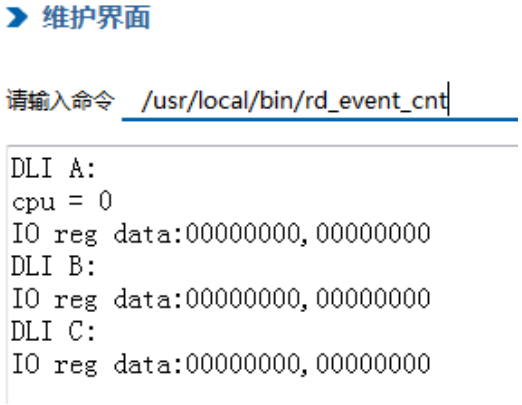


图 5.13 DLI CRC 次数

4. 2. 5　rdmesg

rdmesg 命令打完回车后等待 5s 左右，对话框里显示出所有内核打印信息。

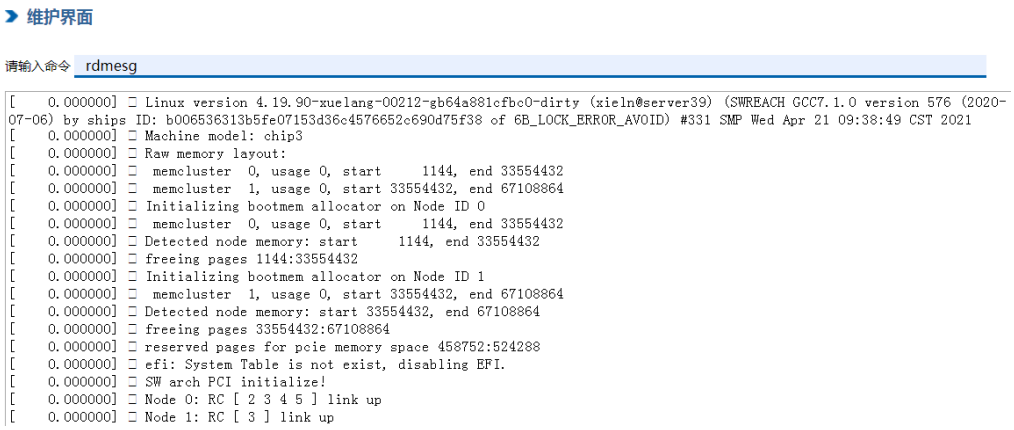


图 5.14 rdmesg

4. 2. 6 scan_error

scan_error 命令用来扫描本机器的所有核心是否存在 fatal error。也可加入参数，扫描指定核心：scan_error [cpu_number]。

无 fatal error:

```
维护界面

请输入命令 scan_error 0

□[0;31;47m-------------------- HMCODE VERSION --------------------□[0m
HMCODE for SW3231 Rev:4cce968
□[0;31;47m-------------------- CORE  0 --------------------□[0m
No fatal error.
```

图 5.15 scan_error(0 核无 error)

若出现 fatal error，则显示错误类型和内部寄存器状态，以下为图例：

```
维护界面

请输入命令 scan_error 0

□[0;31;47m-------------------- HMCODE VERSION --------------------□[0m
HMCODE for SW3231 Rev:4cce968
□[0;31;47m-------------------- CORE  0 --------------------□[0m
ERROR_TYPE                : 0x01 ERROR_DTBM_DOUBLE
ERROR_REASON              : 0x0000000000000000
CSR_IVA_FORM              : 0x0000000000000000
CSR_PTBR                  : 0x0000000000000000
CSR_II_ER                 : 0x0000000000000000
CSR_IIO                   : 0x0000000000000000
CSR_EXC_SUM               : 0x0000000000000000
CSR_EXC_PC                : 0x0000000000000000
CSR_PRI_BASE              : 0x0000000000000000
CSR_IS_CTL                : 0x0000000000000000
CSR_ISSUE_CTL             : 0x0000000000000000
CSR_IS_STAT               : 0x0000000000000000
CSR_VPT_BASE              : 0x0000000000000000
CSR_VPCR                  : 0x0000000000000000
CSR_IA_MATCH              : 0x0000000000000000
CSR_IA_MASK               : 0x0000000000000000
```

图 5.16 scan_error(0 核 error)

若不指定核心，则默认扫描全部核心状态：

```
维护界面

请输入命令 scan_error

□[0;31;47m-------------------- HMCODE VERSION --------------------□[0m
HMCODE for SW3231 Rev:4cce968
□[0;31;47m-------------------- CORE  0 --------------------□[0m
No fatal error.
□[0;31;47m-------------------- CORE  1 --------------------□[0m
No fatal error.
□[0;31;47m-------------------- CORE  2 --------------------□[0m
No fatal error.
□[0;31;47m-------------------- CORE  3 --------------------□[0m
No fatal error.
□[0;31;47m-------------------- CORE  4 --------------------□[0m
No fatal error.
□[0;31;47m-------------------- CORE  5 --------------------□[0m
No fatal error.
□[0;31;47m-------------------- CORE  6 --------------------□[0m
No fatal error.
□[0;31;47m-------------------- CORE  7 --------------------□[0m
No fatal error.
□[0;31;47m-------------------- CORE  8 --------------------□[0m
No fatal error.
```

图 5.17 scan_error(扫描全部核)

4. 2. 7　scan_vcpucb

scan_vcpucb 用来扫描 CPU 的信息，使用时需加入参数，扫描指定核心。

scan_vcpucb　[cpu_number]

维护界面

请输入命令　scan_vcpucb 0

```
□[0;31;47m-------------------- CORE  0 --------------------□[0m
VC__GO_FLAG                 : 0x0000000000000000
VC__PCBB                    : 0x0000000001d4c000
VC__KSP                     : 0xffffffff81d4fe58
VC__USP                     : 0x0000000000000000
VC__KGP                     : 0xffffffff81de9e88
VC__ENT_ARITH               : 0xffffffff80910d80
VC__ENT_IF                  : 0xffffffff80910ec0
VC__ENT_INT                 : 0xffffffff80910d00
VC__ENT_MM                  : 0xffffffff80910e00
VC__ENT_SYS                 : 0xffffffff80911190
VC__ENT_UNA                 : 0xffffffff80910f40
VC__STACK_PC                : 0xffffffff809112c2
VC__NEW_A0                  : 0x0000000000000009
VC__NEW_A1                  : 0x0000000000000000
VC__NEW_A2                  : 0x0000000000000001
VC__WHAMI                   : 0x0000000000000000
VC__CSR_SAVE                : 0x0000000000030000
VC__WAKEUP_MAGIC            : 0x0000000000000000
VC__HOST_VCPUCB             : 0x0000000000000000
VC__UPCR                    : 0x0000000000000000
VC__VPCR                    : 0x0000000000000000
VC__DTB_PCR                 : 0x0000000000000000
VC__GUEST_KSP               : 0x0000000000000000
VC__GUEST_USP               : 0x0000000000000000
VC__VCPU_IRQ_DISABLED       : 0x0000000000000000
VC__VCPU_IRQ                : 0x0000000000000000
VC__PTBR                    : 0x0000000000000000
VC__INT_STAT0               : 0x0000000000000000
VC__INT_STAT1               : 0x0000000000000000
VC__INT_STAT2               : 0x0000000000000000
VC__INT_STAT3               : 0x0000000000000000
VC__RESET_ENTRY             : 0x0000000000000000
VC__PVCPU                   : 0x0000000000000000
VC__EXIT_REASON             : 0x0000000000000000
VC__IPADDR                  : 0x0000000000000000
VC__VCPU_IRQ_VECTOR         : 0x0000000000000000
```

图 5.18 scan_vcpucb

4. 2. 8　查看 CPU 的 PCIE link 状态

维护界面

请输入命令 pcie_link_stat -cpu 0

```
****************PCIE RC0链路状态****************
RC0 link up 失败!
****************PCIE RC2链路状态****************
RC2 link up 成功!
协商的链接宽度:x1
链路速率:Gen1-2.5Gpbs
****************PCIE RC3链路状态****************
RC3 link up 成功!
协商的链接宽度:x4
链路速率:Gen1-2.5Gpbs
****************PCIE RC4链路状态****************
RC4 link up 成功!
协商的链接宽度:x2
链路速率:Gen2-5.0Gbps
****************PCIE RC5链路状态****************
RC5 link up 成功!
协商的链接宽度:x1
链路速率:Gen1-2.5Gpbs
```

图 5.19 CPU 的 PCIE link 状态

4.2.9 查看 rob 超时

维护界面

请输入命令 rob_timeout -w

```
cpu0 core:0,rob_time_out:0x0
cpu0 core:1,rob_time_out:0x0
cpu0 core:2,rob_time_out:0x0
cpu0 core:3,rob_time_out:0x0
cpu0 core:4,rob_time_out:0x0
cpu0 core:5,rob_time_out:0x0
cpu0 core:6,rob_time_out:0x0
cpu0 core:7,rob_time_out:0x0
cpu0 core:8,rob_time_out:0x0
cpu0 core:9,rob_time_out:0x0
cpu0 core:10,rob_time_out:0x0
cpu0 core:11,rob_time_out:0x0
cpu0 core:12,rob_time_out:0x0
cpu0 core:13,rob_time_out:0x0
cpu0 core:14,rob_time_out:0x0
cpu0 core:15,rob_time_out:0x0
cpu0 core:16,rob_time_out:0x0
cpu0 core:17,rob_time_out:0x0
cpu0 core:18,rob_time_out:0x0
cpu0 core:19,rob_time_out:0x0
cpu0 core:20,rob_time_out:0x0
cpu0 core:21,rob_time_out:0x0
```

图 5.20 rob 超时

4.3 其他辅助工具

4.3.1 Rio

rio　<-a address>　　<-cpu cpuid>

说明：

-a ：寄存器地址，见 3231 寄存器手册

-cpu：cpuid=0，选 cpu0，cpuid=1，选 cpu1

示例：

```
请输入命令　/usr/local/bin/rio -a 0x803000000700 -cpu 0
```

```
Read IO reg fuction
cpu num is 0
addr = 0x803000000700
00000000,00000022
```

维护界面

```
请输入命令　/usr/local/bin/rio -a 0x903000000700 -cpu 1
```

```
Read IO reg fuction
cpu num is 1
addr = 0x903000000700
00000000,000000b2
```

图 5.21　rio 示例

4. 3. 2　wio

wio　<-a address>　　<-v value>　<-cpu cpuid>

说明：

-a ：寄存器地址，见 3231 寄存器手册

-cpu：cpuid=0，选 cpu0，cpuid=1，选 cpu1

示例：

维护界面

```
请输入命令　wio -a 0x803000007f00 -v 0x1122334455667788 -cpu 0
```

```
cpu num is 0
write success!
```

维护界面

```
请输入命令  rio -a 0x803000007f00 -cpu 0
```

```
Read IO reg fuction
cpu num is 0
addr = 0x803000007f00
11223344,55667788
```

维护界面

```
请输入命令  wio -a 0x903000007f00 -v 0x1122334455667788 -cpu 1
```

```
cpu num is 1
write success!
```

维护界面

```
请输入命令  rio -a 0x903000007f00 -cpu 1
```

```
Read IO reg fuction
cpu num is 1
addr = 0x903000007f00
11223344,55667788
```

图 5.22 wio 示例

4. 3. 3　rmem

rmem [-cache] [-bwdqA] <-a address> [-l length]

-cache：可 cache 方式读，缺省为不可 cache 方式读

-bwdqA：以字节/字/双字/四倍字/ASCII 方式显示读出结果，缺省为双字方式。

-a address：指定主存起始地址

-l length：指定读主存的长度，缺省为 128 字节。

维护界面

```
请输入命令  /usr/local/bin/rmem -cache -q -a 0x0 -l 0x100
```

```
Usage(v1.0) : rmem [-cache] [-bwdqA] <-a address> [-l length] [-f filename]
0x000000000000: 009ed8bea4f8ec03 025a9cbea4f8b840 0224080000000040 0000000000000140
0x000000000020: 0255480000000100 0000000000000040 0000000000004800 0250004001dd4000
0x000000000040: 0264fc4001db8840 0000000000004840 02500040025af800 0006064002554840
0x000000000060: 3a3130bea4f96440 01e3a80000000132 01e64040070a5c40 ea6495bea4f96440
0x000000000080: 01db88000000010b 01e6404007200c40 06d8b50000001540 0255480000000000
0x0000000000a0: 0007884000035c40 0000070000001640 0000004002554800 00b8144000000000
0x0000000000c0: 0000014002240840 0000000000000100 000000bea4f8e000 01b048400193bc00
0x0000000000e0: 0193bc4002589840 fffde04001b04840 a4f95440025898ff 000000bea4f984be
```

图 5.23 rmem

4. 3. 4　rflash

rflash -a 0x0 -l 0x100 -b　-s 1 -cpu 0

-a　address

-l　lenth

-b　byte

-s　1--flash0　　2--flash1　（默认为 1）

-cpu　0--cpu0 1--cpu1 2--cpu2 3--cpu3

4. 4　固件更新

4. 4. 1　固件烧写

在“固件信息”界面中选择 CPU0/CPU1/ CPU2/CPU3 进行对应的固件烧写，现在界面支持 web 界面烧写 srom、bios、hmcode、DLI 文件、DDR 文件、SAT&DAT 文件、loadfile。点击浏览，选择对应的文件，点击烧写即可。

图 5.24 固件烧写

4. 4. 2　读取固件版本信息

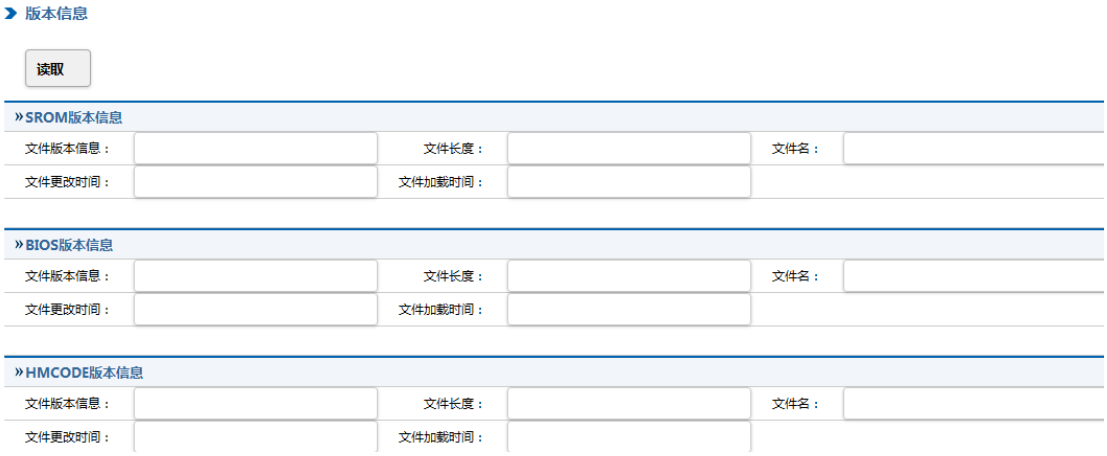


图 5.25　读取固件信息

4. 5　示例

改 4 存控启动：

修改 mc_online 信息，在网页修改即可。

图 5.26　示例

mc_online=0x11　　MC0 4

mc_online=0x55　　MC0 1 4 5

mc_online=0x77　　MC0 1 2 4 5 6

mc_online=0xFF　　MC0 1 2 3 4 5 6 7

更换 SATDAT 文件，在以下烧写 SAT 和 DAT 文件位置选取文件，点击烧写。

图 5.27 示例

4.6 约定

系统和软件使用时，请严格参考本文档所给出的示例。

4.7 处理规程

若用户在使用软件过程中遇到异常状况，不能恢复时，可尝试重启，若无法恢复，请保留现场，联系相关技术人员。

4.8 常见问题

登陆界面后，发现维护功能不见了，可 ssh 登陆小卡（ssh sysadmin@bmc 密码 superuser），使用命令 cat /conf/license/count_days， 若只为 0，则小卡授权过期。

处理方法：寄回先进技术研究院，重新授权。。

5 附录一：pc 值与板级状态对应关系

V6.0.0.d SROM 版本对应 PC 值说明：（和 srom 版本有关，v6.4 和 v6.5 差不多）

Pcie初始化：	0x1448 - 0x1c8c

DLI初始化：	0x1c90 - 0x30f4
DDR 初始化：	0x30f8 - 0x50c4
内存自测试：	0x50c8 - 0x5720
搬 hmcode 和 bios 到主存：	0x5724 - 0x58bc
Hmcode	0x10000 往后

Pc 停在 0x3248，标志位 0x3000，没有读到 SPD 信息

Pc 停在 0x50c4，标志位 0x3006，cpu0 有内存条没有通过训练

Pc 停在 0x13e4，标志位 0x3011，剩下 cpu 有内存条没有通过训练

Pc 停在 0x5720，标志位 0x3014，内存测试没通过

标志位是使用的 MCU_DEBUG3（0x803000004680）

标志位	状态
0x1001	开始加载 sram
0x2000	开始 dli 初始化
0x2001	开始重置 MCU of DLIx PHY
0x2005	开始配置 SerDes
0x2014	Dli 初始化结束
0x3001	确定 1dpc/2dpc
0x3005	固件加载完成
0x300c	开始清主存
0x3011	开始同步
0x3013	开始配置 sat&dat
0x3014	开始 mem test
0x4000	开始加载 hmcode&bios
0x4001	开始同步

6 附录二：基本查错的方法

➢ pc 停在 DLI 初始化查错流程。

（1）判断 Valid Lane

CPU0：

```
rio -a 0x802010000300   -cpu 0      (DLIA)
rio -a 0x802110000300   -cpu 0      (DLIB)
rio -a 0x802210000300   -cpu 0      (DLIC)
```

CPU0：

```
rio -a 0x902010000300   -cpu 1      (DLIA)
rio -a 0x902110000300   -cpu 1      (DLIB)
rio -a 0x902210000300   -cpu 1      (DLIC)
```

正确结果：0x1ff01ff

（2）判断每一条 lane 是否连接

CPU0 DLIA：

```
rio -a 0x80202000c000 -cpu 0
rio -a 0x80202002c000 -cpu 0
rio -a 0x80202004c000 -cpu 0
```

rio -a 0x80202006c000 -cpu 0
rio -a 0x80202008c000 -cpu 0
rio -a 0x8020200ac000 -cpu 0
rio -a 0x8020200cc000 -cpu 0
rio -a 0x8020200ec000 -cpu 0
rio -a 0x80202010c000 -cpu 0

CPU0 DLIB：

rio -a 0x80212000c000 -cpu 0
rio -a 0x80212002c000 -cpu 0
rio -a 0x80212004c000 -cpu 0
rio -a 0x80212006c000 -cpu 0
rio -a 0x80212008c000 -cpu 0
rio -a 0x8021200ac000 -cpu 0
rio -a 0x8021200cc000 -cpu 0
rio -a 0x8021200ec000 -cpu 0
rio -a 0x80212010c000 -cpu 0

CPU0 DLIC：

rio -a 0x80222000c000 -cpu 0
rio -a 0x80222002c000 -cpu 0
rio -a 0x80222004c000 -cpu 0
rio -a 0x80222006c000 -cpu 0
rio -a 0x80222008c000 -cpu 0
rio -a 0x8022200ac000 -cpu 0
rio -a 0x8022200cc000 -cpu 0
rio -a 0x8022200ec000 -cpu 0
rio -a 0x80222010c000 -cpu 0

CPU1 DLIA：

rio -a 0x90202000c000 -cpu 1
rio -a 0x90202002c000 -cpu 1
rio -a 0x90202004c000 -cpu 1
rio -a 0x90202006c000 -cpu 1
rio -a 0x90202008c000 -cpu 1
rio -a 0x9020200ac000 -cpu 1
rio -a 0x9020200cc000 -cpu 1
rio -a 0x9020200ec000 -cpu 1
rio -a 0x90202010c000 -cpu 1

CPU1 DLIB：

rio -a 0x90212000c000 -cpu 1
rio -a 0x90212002c000 -cpu 1
rio -a 0x90212004c000 -cpu 1
rio -a 0x90212006c000 -cpu 1
rio -a 0x90212008c000 -cpu 1
rio -a 0x9021200ac000 -cpu 1
rio -a 0x9021200cc000 -cpu 1

rio -a 0x9021200ec000 -cpu 1
rio -a 0x90212010c000 -cpu 1
CPU1 DLIC：
rio -a 0x90222000c000 -cpu 1
rio -a 0x90222002c000 -cpu 1
rio -a 0x90222004c000 -cpu 1
rio -a 0x90222006c000 -cpu 1
rio -a 0x90222008c000 -cpu 1
rio -a 0x9022200ac000 -cpu 1
rio -a 0x9022200cc000 -cpu 1
rio -a 0x9022200ec000 -cpu 1
rio -a 0x90222010c000 -cpu 1
正确结果：每条 lane 都的值都为 1b’1

- pc 在 DDR 初始化

读 soft_inf 寄存器，若训练没过，读出的值为 0x11c 或者 0x12c；训练通过则为 0x12d。

CPU0：
MC0：rio -a 0x803000007f00
MC1：rio -a 0x803000007f80
MC2：rio -a 0x803000008000
MC3：rio -a 0x803000008080
MC4：rio -a 0x803000008100
MC5：rio -a 0x803000008180
MC6：rio -a 0x803000008200
MC7：rio -a 0x803000008280
训练没过的可以考虑更换内存条或者降频。
CPU1:
MC0：rio -a 0x903000007f00 –cpu 1
MC1：rio -a 0x903000007f80 –cpu 1
MC2：rio -a 0x903000008000 –cpu 1
MC3：rio -a 0x903000008080 –cpu 1
MC4：rio -a 0x903000008100 –cpu 1
MC5：rio -a 0x903000008180 –cpu 1
MC6：rio -a 0x903000008200 –cpu 1
MC7：rio -a 0x903000008280 –cpu 1

- pc 在内存自测试

通过 rmem –a 0xb000000，可以看出自测试出错位置。
pc 在 hmcode 及以后
读总错寄存器：可判断存控多错，LCPM 错误，核心错误等（具体查看 IO 寄存器手册的 DUAL_CGx_FAULT 寄存器）
CPU0：

MC0：rio －a 0x803000006d80
MC1：rio －a 0x803000006e00
MC2：rio －a 0x803000006e80
MC3：rio －a 0x803000006f00
MC4：rio －a 0x803000006f80
MC5：rio －a 0x803000007000
MC6：rio －a 0x803000007080
MC7：rio －a 0x803000007100
CPU1：
MC0：rio －a 0x903000006d80 －cpu 1
MC1：rio －a 0x903000006e00 －cpu 1
MC2：rio －a 0x903000006e80 －cpu 1
MC3：rio －a 0x903000006f00 －cpu 1
MC4：rio －a 0x903000006f80 －cpu 1
MC5：rio －a 0x903000007000 －cpu 1
MC6：rio －a 0x903000007080 －cpu 1
MC7：rio －a 0x903000007100 －cpu 1