申威 1621 维护

接口协议

2017 年 10 月

成都申威科技有限责任公司

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

免责声明

本文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 1621 维护接口协议》主要描述了申威 1621 处理器维护功能说明、维护控制引脚说明、维护接口说明、维护命令详解、JTAG 操作说明等内容。

文档修订

文档更新记录	文档名	申威 1621 维护接口协议
	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 维护功能说明

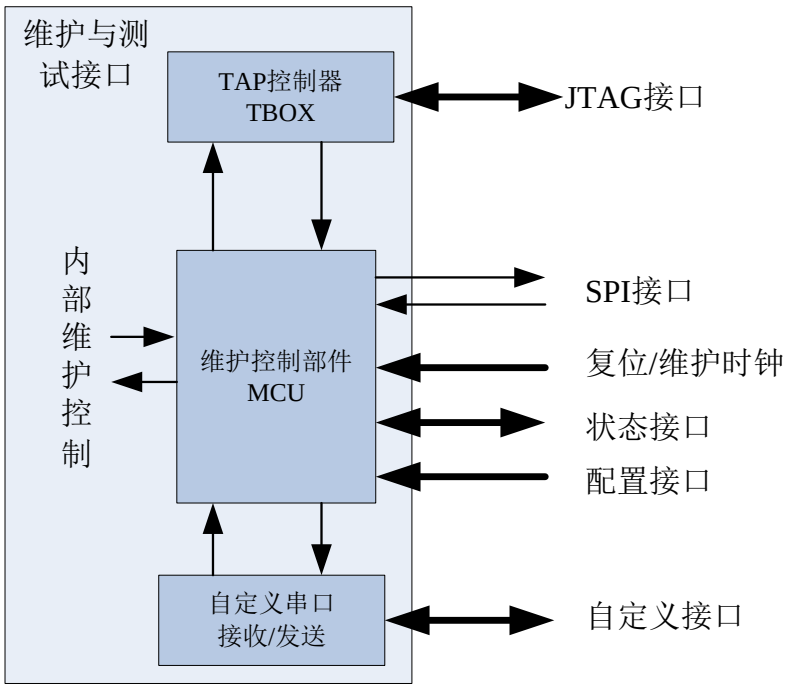

图 1-1：6A 维护控制部件结构

6A 芯片内部的维护控制部件（MCU）控制 6A 芯片的上电复位、冷复位、配置参数加载、内部存储器自测试、初始化程序加载、启动运行、睡眠唤醒、核心断连、运行状态监测。

6A 芯片实现基于标准 SPI 接口的自启动（6A 芯片仅支持作为 SPI Master，连接 SPI Flash 实现自引导）。此外，6A 支持标准 JTAG 接口，除实现对 6A 的测试功能外（包括边界扫描测试和内部扫描测试），也支持基于 JTAG 的的维护操作。

基于上述维护接口，通过维护命令实现对 6A 的维护操作，具体包括：

1)6A 内部 I/O 寄存器读写；

2)6A 控制的存储器读写；

3)内部扫描链寄存器读写；

4)状态监测；

5)针对核心的维护中断，并在此基础上实现的更多维护调试功能；

6)针对核心的睡眠中断和唤醒中断。

2 维护控制引脚

2.1 引脚说明

维护控制相关的 I/O 信号引脚具体如下：

表 2-1：维护控制部件引脚说明

名称	IO	描述	引脚连接关系	备注
Global 信号				
DCOK_H	输入	电源准备好，即上电复位信号（低电平有效）。	Yes 连接 PCB	
RESET_L	输入	冷复位信号，低电平有效。		
MT_CLK_H	输入	维护时钟，由外部系统提供，最大时钟频率为 25MHz。		
自定义串口				
MT_TX_H	输出	串行移位输出。同步于维护时钟，在维护时钟下降沿变化。	不再作为 pin 在封装基板上对 MT_RX_H 接地，MT_RX_H 悬空	1
MT_RX_H	输入	串行移位输入，同步于维护时钟，在维护时钟下降沿变化。		
观测信号				
STAT_OUT_SEL_L [1:0]	输入	状态输出选择，低电平有效。	Yes 连接 PCB	
STAT_OUT_L[4:0]	输出	状态输出，低电平有效。		
时钟控制				
CFG_CCORE_H [4:0]	输入	核心时钟的 PLL 配置。	Yes 连接 PCB	
CFG_XBX_H[3:0]	输入	互连时钟的 PLL 配置。		
CFG_MM_H[3:0]	输入	存控时钟的 PLL 配置。		
初始化模式				
INIT_MODE_H [1:0]	输入	初始化模式配置。	Yes 连接 PCB	

CFG_SEL_L[1:0]	输入	配置方式选择。 “2’b00”：**采用引脚配置方式**，即根据时钟控制引脚和初始化模式引脚来配置 4A； “2’b01”：**采用维护配置方式**，即上述时钟控制引脚和初始化模式引脚无效，采用缺省配置，并可以通过维护命令进行配置。 “2’b1x”：**采用 SPI Master 接口自启动方式**，即自动读取 Flash 获得配置参数。此时 SPI_SEL_L 一定为“1”，SPI_SEL_L 为零时，该配置没有意义。	Yes 连接 PCB	
中断信号				
WAKEUP_L	输入	唤醒中断，默认低电平有效（有效方式可通过 MCU 寄存器进行配置）。	Yes 连接 PCB	
NMI_L	输入	不可屏蔽中断，默认低电平有效（有效方式可通过 MCU 寄存器进行配置）。		
SPI 接口				
MOSI	输入	SPI 接口串行移入	Yes 连接其它芯片	2
MISO	输出	SPI 接口串行移出		
SS_N	输出	选择信号，低有效。		
SCK	输出	时钟信号。最高频率 25MHz。		

注 1：MT_TX_H/ MT_RX_H 仍作为芯片 Bump（SOC 层端口）用于模拟和 FPGA 调试，但实际芯片在封装上接零（不再作为 pin），不再支持自定义串口；

注 2：去掉 CHIP_MODE_H[1:0]不再作为芯片 Bump（SOC 层端口），去掉 SPI_SEL_L 不再作为芯片 Bump（SOC 层端口，MCU 顶层可以接“1”表示使用 Master），SCK/ SS_N 不再作为双向引脚，只作为输出；

注 3：上述引脚 IO Buffer 选择 1.8v LVCMOS；

注 4：状态接口信号可直观反映 6A 的基本运行状态，系统可选择使用，也可连接发光二极管来直观显示运行状态；

注 5：状态接口和配置接口的输入信号支持内部“上拉电阻”，系统不连接时相当于“高电平”。

表 2-2：CFG_SEL_L 说明

CFG_SEL_L[1:0]	具体含义
2’b00	**引脚配置方式。** 初始化模式（INIT_MODE）、时钟配置（CFG_xx_H）由芯片引脚控制，并可以由维护命令进行配置。
2’b01	**维护命令配置方式。** 初始化模式（INIT_MODE）、时钟配置（CFG_xx_H）采用缺省配置，并可以由维护命令进行配置。 **该配置主要用于对 SPI Flash 的在线烧录。**
2’b10	**自引导方式。** 初始化模式（INIT_MODE）、时钟配置（CFG_xx_H）来自 SPI Flash 的配置信息（硬件自动读取）。
2’b11	

说明：SPI_SEL_L 不在作为芯片 bump，在 MCU 内部接“1”，即只支持 Master。此时 CFG_SEL_L[1:0]配置为 2’b01，可在主状态机的配置 1 状态对 Flash 进行烧录。正常的使用模式 CFG_SEL_L[1:0]配置为 2’b1x，实现自启动。

2.2 初始化状态机

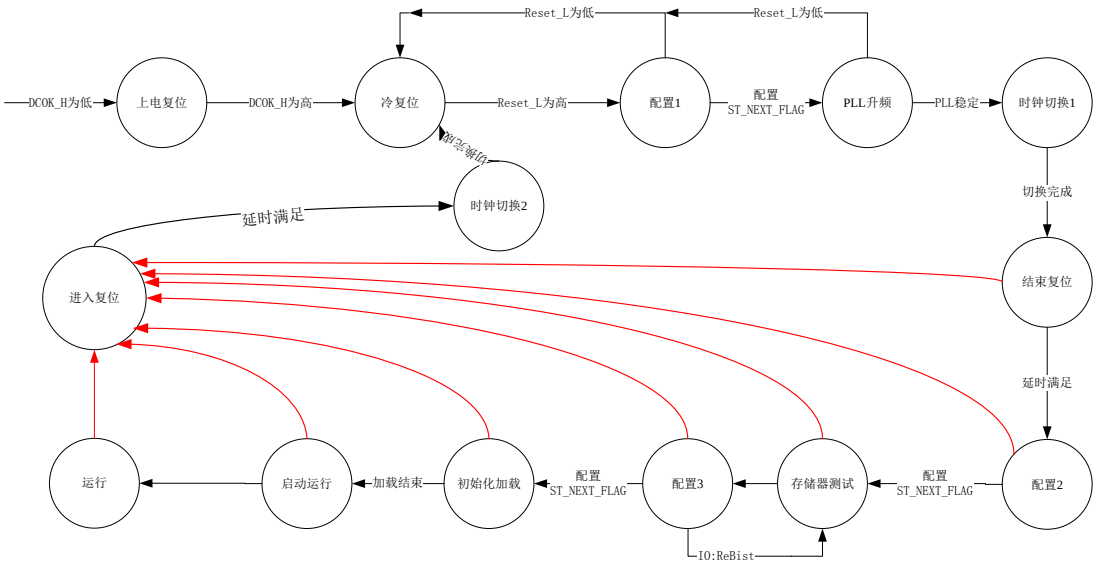

图 2-1：6A 初始化状态机

1)上电复位（4’b0000）：若引脚 DOCK_H 为“0”，则无条件处于此状态。此状态下，若 DCOK_H 变为“1”，则转向冷复位状态；

2)冷复位（4’b0001）：若引脚 RESET_L 为“0”，则维持在此状态，否则转向配置 1 状态；**注：JTAG 接口（基于 JTAG 接口的特殊维护命令）增加冷复位，冷复位有效（冷复位有效 1000 个维护时钟），则进行冷复位流程。注：6A 芯片不在支持 WatchDog 的冷复位。**

3)配置 1（4’b0010）：如果配置 1 状态使能寄存器 **CFG1_EN** 配置为零则直接转向 PLL 升频状态，否则等待初始化相关的控制寄存器完成配置后，通过写**“ST_NEXT_FLAG”**寄存器的维护命令，则转向 PLL 升频状态。引脚 CFG_SEL_L 为“2’b01”时，通过维护命令配置初始化相关的控制寄存器。引脚 CFG_SEL_L 为“2’b00”时，根据配置引脚 INIT_MODE_H 配置初始化相关的控制寄存器；引脚 CFG_SEL_L 为“2’b1x”时，则自动读取 Flash 并配置初始化相关的控制寄存器；**状态机时实现时，根据 IOR 的配置（INIT_CTL[CFG1_EN]）进行自动跳转或等待软件写 ST_NEXT_FLAG。CFG1_EN 默认为 1，会根据配置引脚 INIT_MODE_H、维护操作或自动参数读取配置该值触发状态机跳转。**

4)PLL 升频（4’b0011）：如果参数配置寄存器设置成 PLL 旁路模式（**PLLUP_EN** 为零），则直接转入时钟切换状态；否则通过计数器计数方式等待 PLL 升频稳定，再转入时钟切换 1 状态。由于存储器测试状态增加 Page Cache 测试（PIU 的 IOMMU 相关），而 Page Cache 的 Clock 来自 PCI-E PHY，所以此处需要等待 PCI-E PHY 的 Clock 稳定。**硬件实现时在 MCU 设置计数器用于等待 PClk 稳定，同时监测来自 PCI-E PHY 的 Clock Stable 信号；**

5)时钟切换 1（4’b0100）：按顺序进行工作时钟的高低频切换。完成频率切换后，转向结束复位状态；

6)结束复位（4’b0101）：结束 6A 的复位，等待一段时间后，转向配置 2 状态；

7)配置 2（4’b0110）：如果参数配置寄存器（**CFG2_EN 为零**）设置成跳过此状态，则直接转向存储器自测试状态，否则外部先使用维护命令设置与存储器自测试相关的扫描链寄存器，然后向 6A 发出写“**ST_NEXT_FLAG**”寄存器的维护命令，则转向存储器自测试状态；**状态机时实现时，根据 IOR 的配置（INIT_CTL[CFG2_EN]）进行自动跳转或等待软件写 ST_NEXT_FLAG。CFG2_EN 默认为 1，会根据引脚、维护操作或自动参数读取配置该值触发状态机跳转。自启动模式会把 CFG2_EN 配置为零，对该 IOR 的隐式写在配置 1 状态已经完成。**

8)存储器自测试（4’b0111）：如果参数配置寄存器（BIST_EN 为零）设置成跳过此状态，则直接转向配置 3 状态，否则对内部存储器进行自测试，存储器自测试结束转向配置 3 状态；如果等待指定时间尚未收到存储器自测试结束标志，则作为超时而转向配置 3 状态；

9)配置 3（4’b1000）：如果参数配置寄存器（**CFG3_EN 为零**）设置成跳过此状态，则直接转向初始化加载状态，否则外部首先使用维护命令设置 DDR3 存储器接口以及 PCI-E 接口等相关 I/O 寄存器，然后向 6A 发出写“ST_NEXT_FLAG”寄存器的维护命令，则转向初始化

加载状态，外部向 6A 发出“重新存储器自测试”维护命令，则转向存储器自测试状态；**状态机时实现时，根据 IOR 的配置（INIT_CTL[CFG3_EN]）进行自动跳转或等待软件写 ST_NEXT_FLAG。CFG3_EN 默认为 1，会根据引脚、维护操作或自动参数读取配置该值触发状态机跳转。自启动模式会把 CFG3_EN 配置为零，对该 IOR 的隐式写在配置 1 状态已经完成。**

10)初始化加载（4’b1001）：如果参数配置寄存器（ICLD_EN 为零）设置成跳过此状态，则直接转向启动运行状态，否则外部首先使用维护命令将初始化程序加载到**核心**的指令 Cache 中，然后外部向 6A 发出“初始化加载结束”的维护命令，则转向启动运行状态；**自启动模式自动读取 Icache 内容，并进行加载，加载结束后（32KB）进行状态机跳转。**

11)启动运行（4’b1010）：如果参数配置寄存器（STRUN_EN 为零）设置不启动运行，则维持在该状态，否则转入运行状态。**主状态机转入运行状态时，会启动各核心子状态机（从 IDLE 状态转出）。**

12)运行（4’b1011）：运行状态；

13)进入复位（4’b110x）：在结束复位状态、配置 2 状态、存储器自测试状态、配置 3 状态、初始化加载状态、启动运行状态和运行状态下，若引脚 Reset_L 变为“0”（或者来自 JTAG 的冷复位命令），则转向此状态。在该状态下，使得 6A 中除维护控制部件和 IOR 之外的其它所有部件进入复位状态，等待一段时间好后，转向时钟切换 2 状态；

14)时钟切换 2（4’b111x）：按顺序进行各模块的高低频工作时钟切换，然后转向冷复位状态。

注 1：配置 1 状态下外部只能读写维护控制部件的 I/O 寄存器，配置 2 状态和配置 3 状态下可以修改任意的 I/O 寄存器，但配置 2 状态主要用于修改与存储器自测试相关的扫描链寄存器。

2.3 参数配置寄存器

参数配置寄存器是维护控制部件对 6A 进行主要配置的 I/O 寄存器，此寄存器可通过复位（默认值）、引脚（**CFG_SEL_L[1:0]为 2’b00**）和维护命令（**CFG_SEL_L[1:0]为 2’b01**）进行设置，也可以根据 Falsh 信息进行自引导（**CFG_SEL_L[1:0]为 2’b1x**），核心只可以读此寄存器内容，但不能修改。该寄存器内容如下：

1)寄存器[4:0]位：为核心频率配置 CCORE_FREQ[4:0]，用于配置产生核心工作时钟的 PLL。在配置 1 状态下，若引脚 CFG_SEL_L 为“0”，则修改为引脚 CFG_CORE_H[4:0]的值，否则采用缺省值或在此状态下通过外部维护命令进行修改；

2)寄存器[11:8]位：为互连频率配置 XBX_FREQ[3:0]，用于配置产生互连（包括 CPM、IRU 和 INTPU）工作时钟的 PLL。在配置 1 状态下，若引脚 CFG_SEL_L 为“0”，则修改为引脚 CFG_XBX_H[3:0]的值，否则采用缺省值或在此状态下通过外部维护命令进行修改；

3)寄存器[19:16]位：为存控频率配置 MC_FREQ[3:0]，用于配置产生存控（MC）工作时钟的 PLL。在配置 1 状态下，若引脚 CFG_SEL_L 为“0”，则修改为引脚 CFG_MM_H[3:0]的值，否则采用缺省配置或在此状态下通过外部维护命令进行修改；

4)寄存器[41:32]位：为复位初始化控制 INIT_CTL[9:0]，用于配置初始化状态机的状态转换方式，从低到高位，分别对应配置 1 状态、PLL 升频状态、时钟切换状态（时钟切换 1/时钟切换 2 使用相同的控制位）、结束复位状态、配置 2 状态、存储器自测试状态、配置 3 状态、初始化加载状态、控制核心运行状态和进入复位状态。若对应状态的控制位为“0”，则在该状态下不进行任何操作而快速结束此状态，或者缩短此状态的处理时间。在冷复位状态下设置为缺省值（10’h3ff）；在配置 1 状态下，若引脚 CFG_SEL_L 为“0”，则根据引脚 INIT_MODE_H[1:0]来设置，否则采用缺省状态或在此状态下通过外部维护命令进行修改；

5)寄存器其它位：保留，写这些位不产生任何作用，读出值为“0”。

2.4 初始化模式

初始化模式配置引脚 INIT_MODE_H[1:0]的各种配置含义如下表所示。

表 2-3：INIT_MODE_H[1:0]说明

<table>
<tr><th>INIT_MODE_H[1:0]
初始化状态机状态</th><th>2’b00</th><th>2’b01</th><th>2’b10</th><th>2’b11</th></tr>
<tr><td>配置 1 状态</td><td colspan="3">无操作</td><td>可读写维护控制器的 I/O 寄存器</td></tr>
<tr><td>PLL 升频状态</td><td colspan="3">无操作</td><td>延时等待</td></tr>
<tr><td>时钟切换状态</td><td colspan="3">快速进行时钟切换</td><td>正常进行时钟切换</td></tr>
<tr><td>结束复位状态</td><td colspan="3">无操作</td><td>延时等待</td></tr>
<tr><td>配置 2 状态</td><td>停在此状态</td><td>无操作</td><td colspan="2">可读写维护控制器的 I/O 寄存器和存储器自测试相关的扫描链寄存器</td></tr>
<tr><td>存储器自测试状态</td><td rowspan="4">可在配置 2 状态下修改参数配置寄存器来选择</td><td>无操作</td><td colspan="2">进行存储器自测试</td></tr>
<tr><td>配置 3 状态</td><td>无操作</td><td colspan="2">可读写所有 I/O 寄存器</td></tr>
<tr><td>初始化加载状态</td><td>无操作</td><td colspan="2">进行初始化程序加载</td></tr>
<tr><td>控制核心运行状态</td><td>停在此状态</td><td colspan="2">延时等待</td></tr>
<tr><td>进入复位状态</td><td colspan="3">无操作</td><td>延时等待</td></tr>
<tr><td>应用场景</td><td>测试模拟</td><td>程序模拟</td><td>运行模拟</td><td>正常运行</td></tr>
</table>

注 1：在引脚 CFG_SEL_L 为“2’b00”时，引脚 INIT_MODE_H[1:0]决定初始化的方式（即根

据 INIT_MODE_H[1:0]配置 IOR 的默认值），其中 2’b00~2’b10 为模拟测试模式，分别用于测试模拟（如存储器自测试）、程序模拟（运行模拟验证程序）和运行模拟（仿真实际运行模式），采用缺省配置加快模拟速度，并缩短部分状态处理的等待时间，如时钟切换间隔在快速模式下为 2 个维护时钟周期，否则缺省为 65535 个维护时钟周期（可通过维护命令设置）。该初始化模式配置下，仍允许通过维护操作修改寄存器值。

注 2：引脚 CFG_SEL_L 为“2’b01”时，可以通过维护操作修改 IOR 值。引脚 CFG_SEL_L 为“2’b1x”时，**硬件会根据初始化模式自动读取配置信息完成初始化流程。**

2.5 状态输出

当 STAT_OUT_SEL_L 为 2’b11 时，STAT_OUT_L[4:0]输出状态所表示的意义如下所示：

表 2-4：维护状态输出说明(1)

域	意义说明
STAT_OUT_L[4]	芯片总错标志（包括维护接口错），**低电平有效**。
STAT_OUT_L[3]	中断完成标志，**低电平有效**。
STAT_OUT_L[2]	存储器自测试完成，**低电平有效**。
STAT_OUT_L[1:0]	存储器自测试结果，含义如下： 2’b00，表示测试无错； 2’b01，表示有错可修复； 2’b10，表示有错不可修复； 2’b11，表示测试时间超时。

当 STAT_OUT_SEL_L 为 2’b10 时，STAT_OUT_L[4:0]输出状态所表示的意义如下所示：

表 2-5：维护状态输出说明(2)

域	意义说明
STAT_OUT_L[4]	自定义串口维护命令错标志，**低电平有效。**
STAT_OUT_L[3:0]	维护主状态机状态（在维护时钟的下降沿输出），具体如下： 4'b0000 DCOKRST 表示上电复位状态（等待 DCOK_H 有效）； 4'b0001 COLDRST 表示冷复位状态（等待 RESET_L 无效）； 4'b0010 CONFIG1 表示配置 1 状态（MCU 配置）； 4'b0011 WAITPLL 表示 PLL 升频状态； 4'b0100 WAITUPCLK 表示时钟切换 1 状态； 4’b0101 SOCRSTEND 表示结束复位状态； 4'b0110 CONFIG2 表示配置 2 状态（存储器自测试配置）； 4'b0111 MEMBIST 表示存储器自测试状态；

	4'b1000 CONFIG3　　表示配置 3 状态（其它接口配置）； 4'b1001 SROMLD　　表示初始化程序加载状态； 4'b1010 STARTRUN　　表示启动运行状态； 4'b1011 RUN　　表示运行状态； 4’b1100 ENTERRST　　表示进入复位状态； 4'b1110 WAITDOWNCLK　表示时钟切换 2 状态。 其它编码保留。

当 STAT_OUT_SEL_L 为 2’b01 时，STAT_OUT_L[4:0]输出状态所表示的意义如下所示：

表 2-6：维护状态输出说明(3)

<table>
<tr><th>域</th><th colspan="2">意义说明</th></tr>
<tr><td>STAT_OUT_L[4]</td><td colspan="2">指示在存储器 Debug 测试方式下，发现错误而暂停存储器 Debug 测试，此时可通过状态扫描获得具体的错误信息，低电平有效。</td></tr>
<tr><td>STAT_OUT_L[3]</td><td>对应特殊的寄存器 FlagReg[3]的非。</td><td rowspan="4">FlagReg[3:0]寄存器核心可读写，通过编码表示各类状态，具体状态信息意义由软件控制。由于电平用以驱动发光二极管，所以 TestOut 状态输出 FlagReg[3:0]寄存器的非（低电平有效）。</td></tr>
<tr><td>STAT_OUT_L[2]</td><td>对应特殊的寄存器 FlagReg[2]的非。</td></tr>
<tr><td>STAT_OUT_L[1]</td><td>对应特殊的寄存器 FlagReg[1]的非。</td></tr>
<tr><td>STAT_OUT_L[0]</td><td>对应特殊的寄存器 FlagReg[0]的非。</td></tr>
</table>

当 STAT_OUT_SEL_L 为 2’b00 时，需要根据 IOR：TESTSEL 选择 STAT_OUT_L[4:0]输出需要观测的内容，意义如下所示：

表 2-7：维护状态输出说明(4)

IOR 寄存器 TESTSEL	**内部信号**	**对应的 TEST_OUT**	**含义说明**
0（默认）	保留	TEST_OUT[0]	保留
	保留	TEST_OUT[1]	保留
	保留	TEST_OUT[2]	保留
	保留	TEST_OUT[3]	保留
	CLU2SI_CCORE_CLKOUT	TEST_OUT[4]	核心 PLL 输出的 32 分频时钟观测信号。
1	保留	TEST_OUT[0]	保留
	保留	TEST_OUT[1]	保留
	保留	TEST_OUT[2]	保留

	保留	TEST_OUT[3]	保留
	CLU2SI_MM_CLKOUT	TEST_OUT[4]	存控 PLL 输出的 16 分频时钟观测信号。
2	保留	TEST_OUT[0]	保留
	保留	TEST_OUT[1]	保留
	保留	TEST_OUT[2]	保留
	保留	TEST_OUT[3]	保留
	CLU2SI_ RING_CLKOUT	TEST_OUT[4]	环网 PLL 输出的 32 分频时钟观测信号。
3	CLU2SI_CORE_LOCK	TEST_OUT[0]	核心 PLL 时钟锁定。
	CLU2SI_MM_LOCK	TEST_OUT[1]	存控 PLL 时钟锁定。
	CLU2SI_ RING_LOCK	TEST_OUT[2]	环网 PLL 时钟锁定。
	保留	TEST_OUT[3]	保留。
	保留	TEST_OUT[4]	保留。
4	PCIE0_CLKOUT_PCI_H	TEST_OUT[0]	PCI-E0 PLL 输出时钟，8 分频输出。
	PCIE0_LINK_UP	TEST_OUT[1]	PCI-E0 接收方链路连接建立。
	PCIE1_CLKOUT_PCI_H	TEST_OUT[2]	PCI-E1 PLL 输出时钟，8 分频输出。
	PCIE1_LINK_UP	TEST_OUT[3]	PCI-E1 接收方链路连接建立。
	CLKOUT_MT	TEST_OUT[4]	维护时钟。
5	RingOut[0]	TEST_OUT[0]	核心 0 的环振测试输出。
	RingOut[1]	TEST_OUT[1]	核心 1 的环振测试输出。
	RingOut[2]	TEST_OUT[2]	核心 2 的环振测试输出。
	RingOut[3]	TEST_OUT[3]	核心 3 的环振测试输出。
	RingOut[4]	TEST_OUT[4]	核心 4 的环振测试输出。
6	RingOut[5]	TEST_OUT[0]	核心 5 的环振测试输出。
	RingOut[6]	TEST_OUT[1]	核心 6 的环振测试输出。
	RingOut[7]	TEST_OUT[2]	核心 7 的环振测试输出。
	RingOut[8]	TEST_OUT[3]	核心 8 的环振测试输出。
	RingOut[9]	TEST_OUT[4]	核心 9 的环振测试输出。
7	RingOut[10]	TEST_OUT[0]	核心 10 的环振测试输出。
	RingOut[11]	TEST_OUT[1]	核心 11 的环振测试输出。
	RingOut[12]	TEST_OUT[2]	核心 12 的环振测试输出。
	RingOut[13]	TEST_OUT[3]	核心 13 的环振测试输出。

	RingOut[14]	TEST_OUT[4]	核心 14 的环振测试输出。
8	RingOut[15]	TEST_OUT[0]	核心 15 的环振测试输出。
	CLU2SI_CFG_MM_H[3:0]	TEST_OUT[4:1]	存控 PLL 配置。
9	CLU2SI_CFG_RING_H[3:0]	TEST_OUT[3:0]	环网 PLL 配置。
	保留	TEST_OUT[4]	保留。
10	CLU2SI_CFG_CORE_H[4:0]	TEST_OUT[4:0]	核心 PLL 配置。
其它	保留	保留	保留。

2.6 维护接口配置

表 2-8：6A 芯片维护接口配置

接口	6A 实现	备注
JTAG 接口维护操作	支持	主要维护通路，通过标准 JTAG 接口（利用转接器连接 PC）实现芯片维护调试。
自定义接口（RX/TX）维护操作	支持（开发阶段使用）	MT_TX_H/ MT_RX_H 仍作为芯片 Bump（SOC 层端口）用于模拟和 FPGA 调试，但实际芯片在封装时不再作为 pin（输入接零，输出悬空），**实际芯片（产品）不再支持自定义串口；**
SPI Master 自引导	支持	连接 SPI Flash 实现自引导。**常规的芯片启动模式。**

3 自定义串行维护接口

3.1 简介

通过自定义串口维护接口（MT_TX_H/MT_RX_H），外部系统维护控制（包括 I/O 套片 ICH）实现维护操作。该接口是基于维护时钟的同步接口，采用下降沿输出、上升沿接收的方式进行容偏斜设计。

基本操作模式是：外部维护控制通过 MT_RX_H 发出维护命令，芯片进行处理后通过 MT_TX_H 返回维护响应。维护命令和维护响应格式见第 5 章。

维护命令和维护响应的传输方式类似，采用包格式，以字节为基本单位，根据具体的维护命令包和维护响应包类型，包含若干字节。传输时，总是首先输出低字节，然后输出高字节，每个字节内首先输出高位，然后输出低位。维护命令包和维护响应包都是按字节进行偶校验。

3.2 处理流程

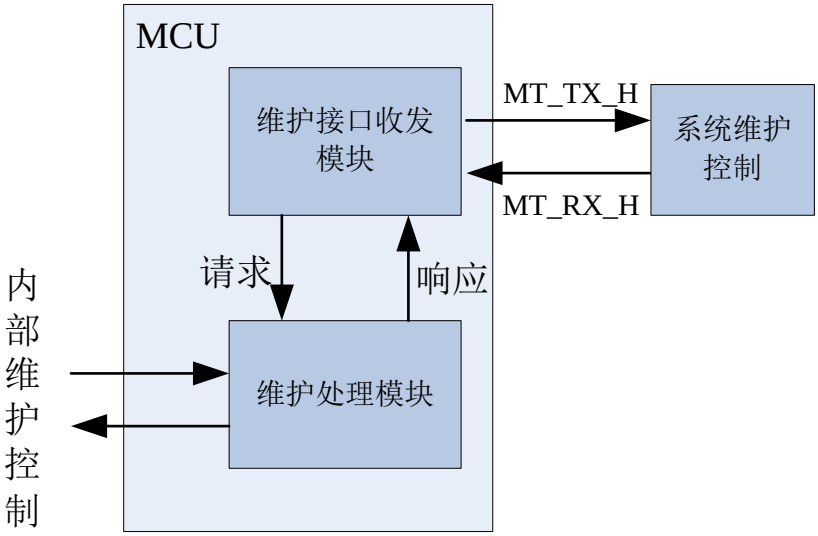

图 3-1：自定义维护接口结构

图 3-1 给出了自定义维护接口的结构图，主要包括维护接口收发逻辑和维护处理逻辑。具体操作流程如下：

1）维护命令接收：在上电复位结束后，维护接口收发模块处于总是可以接收维护命令状态。维护命令接收由 4 状态的状态机控制，IDLE0 状态下收到请求包头（2’b11）第一位则转入 IDLE1 状态，IDLE1 状态下收到请求包头（2’b11）第二位则转入 CMD 状态，CMD 状态下接收命令信息（包括命令、地址、数据有效位和数据。CMD 状态下根据维护命令收齐命令信息后转入 IDLE0 状态，并将有效的维护命令及其附带的数据，发送到维护处理模块；

2）维护请求的处理：维护处理模块根据具体的维护命令，在 MCU 内部处理或者向 6A 内部其它模块转发。收回目标模块产生的处理结果，形成响应后，转发到维护接口收发模块；

3）维护响应发送：在上电复位结束后，维护接口收发模块处于总是可以发送维护响应状态。维护响应发送由 2 状态的状态机控制，IDLE 状态下收到维护处理模块发来的维护响应，则转入 SND

状态，SND 状态下根据维护响应包格式发完响应后，再切换到 IDLE 状态，同时通知维护处理模块。

6A 总是串行处理维护命令，即接受维护命令、处理并发出响应后，才能接受新的维护命令，否则（外部系统发送维护命令违反顺序性）作为一种维护错误处理（维护请求丢失，并登记 MCU 的 IOR：ERR_INF[18]）。收到保留的维护命令或者发现维护命令包出现偶校验错，也作为维护错误处理（返回非法地址响应或者偶校验错响应）。

如果维护命令因各种异常情况出现超时（外部系统判断），系统维护控制可以向 6A 发送“维护接口复位命令”（该命令对外部而言是一条 IO 写请求，6A 向系统返回写结束响应）复位维护处理模块的相关逻辑，使得前面的所有维护命令作废。

特别注意：为保证“维护接口复位命令”能够被 6A 正确处理，要求系统维护控制在发出此命令前的 1024 个维护时钟周期内，不发出任何其它维护命令，此命令发出后，与后继的维护命令间隔不小于 1024 个维护时钟周期。

4 芯片自启动

4.1 简介

通过标准 SPI 接口（6A 芯片作为 Master），外部连接 Flash 芯片（NorFlash，推荐使用 ATMEL AT25DF081），通过读取 Flash 内存储的控制信息和 Srom 加载信息实现芯片自启动。

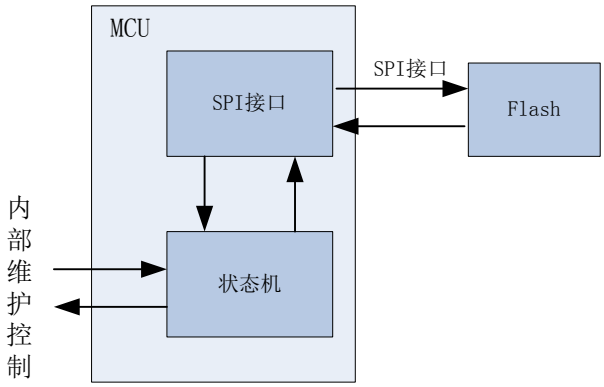

图 4-1：6A 芯片标准 SPI 接口自启动示意图

Flash 对应的存储空间被编址在 MCU 对应的 IO 空间上，维护可以通过自定义接口、JTAG 接口可以实现对 Flash 空间的读写操作，核心也可以进行读写操作。芯片初始化状态机会根据配置对 Flash 进行自动读取。

Flash 控制器支持以下三种读写操作：

1)64bit 读：维护或者核心以 IOR 地址访问 Flash 时，SPI Flash 控制器一次性返回 64bit 数据。

2)1024bit 读：初始化状态机进行 Icache 加载时，SPI Flash 控制器一次性返回一个 Cache 行 1024bit 的数据。

3)64bit 写：Flash 写操作则一律按 64bit 的粒度来。

Flash 控制器支持的擦写粒度有如下 3 种：

1)4K 字节擦除；

2)32K 字节擦除；

3)64K 字节擦除。

4.2 空间划分

Flash 空间划分如下所示：

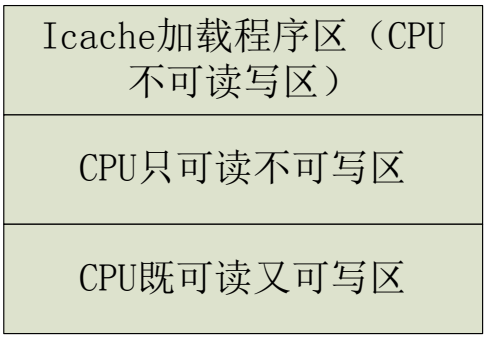

图 4-2：Flash 空间划分

Flash 空间分为以下三个区：

1)ICache 加载程序区大小固定为 32KB（对应 ICache 容量，为最小 Flash 容量。最大 Flash 容量是 16MB）。核心对 ICache 加载程序区不可读也不可写。维护接口对 Flash 所有空间既可读又可写；

2)核心只读区域；区域大小可配（以 32KB 为边界），配置信息由 **IOR：FlashInfo** 给出；

3)核心可读写区域。区域大小可配（以 32KB 为边界），配置信息由 **IOR：FlashInfo** 给出。

对软件来说，Flash 存储空间映射为 MCU 的 IOR（PA[39:36]=4’b1101）。进一步规定 PA[35]=1’b0 表示 MCU 原有的 IOR 空间，PA[35]=1’b1 表示映射到 Flash 的 IOR 空间。维护以及软件均以访问 IOR 的方式对 SPI Flash 进行访问。每一个 IOR 地址对应 Flash 的 64 位数据。

ICache 加载程序区大小固定为 32KB。

1)其中第 0~63bit 为校验和信息（校验和信息是 ICache 加载程序区中除 0~63bit 数据之外的其他所有对界的 64 位数据进行异或而得，即校验和[63:0]= ICache 加载程序区[127:64]^ ICache 加载程序区[191:128]^……^ICache 加载程序区[32*1024-1: 31*1024+960]）,校验和应由维护软件在 Flash 首次编程的时候计算写入。

2)64~1023bit 为 CPU 配置信息，由维护软件在 Flash 首次编程的时候写入，各位具体如下：

◆[127: 64]：写入 IOR：FLASH_INFO（只读寄存器）。包括 Flahs 空间大小、核心只读区域空间大小和核心可读写区域空间大小。根据 IOR 实际范围，64 位中的其它位保留；

◆[191:128]：写入 IOR：CPU_INFO（只读寄存器）。包括 CPU 生产信息。

◆[255:192]：写入 IOR：INIT_CTL（包括时钟配置和初始化控制）。根据 IOR 实际范围，64 位中的其它位保留；

◆[287:256]：写入 IOR：PLL_CHG_CNT。

◆[319:288]：写入 IOR：PIUPLL_CNT。

◆[351:320]：写入 IOR：BIST_CTL。根据 IOR 实际范围，32 位中的其它位保留；

◆其它位保留为零

3)从 1024bit 开始，为 32KB（**特别注意，准确的容量是 32KB 减去一个 Cache 行，为了便于整理此处仍标记为 32KB**）的 ICache 加载程序，由维护软件在 Flash 首次编程的时候写入。若 ICache 加载程序实际大小不满 32KB，剩余部分维护软件应该用停机指令填充。**MCU 在进行 Icache 加载时，在加载完毕上述 ICache 加载程序后，还会补足一个 Cache 行 1024 位的 Halt 指令，以装满 32KB 大小的 Icache。**

4.3 处理流程

主要包括两个基本流程：

1)Flash 信息烧录（出厂设置）：只在芯片出厂时需要烧录。CFG_SEL_L[1:0]配置成 2’b01，通过维护接口（标准 JTAG 接口）对 Flash 空间进行写操作（64b 粒度），完成信息烧录。

2)芯片自启动（常规启动模式）：芯片一般应用模式。CFG_SEL_L[1:0]配置成 2’b1x，芯片初始化状态机（2.2 节）配置 1 状态会自动读取 Flash，并且 Srom 加载状态时会自动读取 Flash。

4.4 SPI Flash 编址说明

6A 芯片使用基于 SPI Flash 的自启动流程，硬件自动读取 SPI Flash 内的配置参数和 Srom 程序。Srom 程序利用访存通路（IO 访问，SPI Flash 编址在 MCU 的 IO 空间，并且 PA[35]=1 表示是 Flash 空间）把 Flash 中的 HMCODE、BIOS 搬家到主存，然后 Srom 程序跳转到指定位置取指。该过程仍类似 410/411 芯片，410/411 芯片是通过 SROM 对 EP 设备的 IO 空间进行读取搬家到内存。

SPI Flash 空间的每个 64 位对应一个 64 位 IO 寄存器（SPI Flash 空间仅支持对界的 8B 访问），由于 MCU 的 IO 空间按 Cache 块编址，所以物理地址{PA[27:7],3’b000}作为 SPI Flash 空间的访问地址（SPI Flash 空间最大支持 16MB）。假设要访问 SPI Flash 的地址 Addr_a（Flash 域），需要把 Addr_a 左移四位（**最低四位补零。假设访问 Flash 的第 2 个 64 位，即 Flash 域地址 0x8，转换为物理地址其低位是 0x80**），并且[38:35]拼上 4’b1011、[47]拼上 1’b1（6A 的物理地址修改为 48 位，由[47]表示 IO 空间或 Mem 空间。特殊说明维护操作仍是 40 位物理地址）得到 IO 访问地址。**无论核心 IO 访问或者维护请求都需要按照该方式生成地址。对于擦除操作指定的地址，也需要按上述方式生成。软件在生成 IO 地址时需要予以关注。**

注：维护操作仍是 40 位物理地址，与 4A/4C 芯片相同。

5 维护命令

5.1 维护命令注意事项

特殊说明：维护命令中的地址仍按照 40 位物理地址进行访问，与 4A/4C 芯片相同。

MCU 对 PIU、IRU、INTPU、CPM 和 MC 的 IO 访问仍按照 40 位地址的接口协议进行，其中[39]含义与 4A 相同。

核心发出的 48 位地址由源方 CPM 进行重新映射，转换为 40 位的地址（原接口协议）发给 CPM、MC、PIU、IRU 和 INTPU。

5.2 维护命令格式

5.2.1 维护命令编码

编号	维护命令	[7:6]	[5:0]	备注
1	维护读存储器	固定为 2’b11	6’b000000	带 CC（一致性处理标志）的存储器读（每次读 128 字节）。
2			6’b000100	不带 CC（一致性处理标志）的存储器读（每次读 128 字节）。
3	维护写存储器		6’b100000	带 CC（一致性处理标志）的存储器写（写数据 16 字节）。
4			6’b100101	带 CC（一致性处理标志）的存储器写（写数据 128 字节）。
5			6’b101000	不带 CC（一致性处理标志）的存储器写（写数据 16 字节）。
6			6’b101001	不带 CC（一致性处理标志）的存储器写（写数据 128 字节）。
7	维护接口复位		6’b110000	维护接口复位。
8	维护读寄存器		6’b001000	每次读 8 字节。
9	维护写寄存器		6’b100100	每次写 8 字节。
10	扫出或监测状态		6’b001100	每次扫出 128 字节。
11	扫入状态		6’b101101	串行扫入（16 字节）。
12			6’b101111	串行扫入（128 字节）。
13	初始化加载		6’b111100	初始化加载（128 字节）。
14			6’b111111	初始化加载结束。

5.2.2 维护中断命令

维护中断命令通过对维护中断控制寄存器（IOR：MT_INT）的写来实现，产生对指定核心的维护中断、睡眠中断或唤醒中断请求。其中第 9 字节的最低 2 位指示中断目标核心（编码表示）。维护中断命令可实现复杂的维护操作，也可对正在运行的核心发睡眠中断或对已经睡眠的核心发唤醒中断。处于断开状态的核心如果要恢复运行状态，则先要修改核心断连标志，然后再发唤醒中断。唤醒后核心从主存 CSR:PRI_BASE＋0x1000 开始执行。

字节 0	2’b11	6’b100100
字节 1	地址[7:0]	
字节 2	地址[15:8]	
字节 3	地址[23:16]	
字节 4	地址[31:24]	
字节 5	地址[39:32]	
字节 6	数据有效位[7:0]	
字节 7	8’b00000000	
字节 8	最高 4 位为全“0”，最低 4 位指示中断的目标核心，	
字节 9	最高 6 位为全“0”，最低 2 位指示中断类型编码	
字节 10	8’b00000000	
字节 11	8’b00000000	
字节 12	8’b00000000	
字节 13	8’b00000000	
字节 14	8’b00000000	
字节 15	8’b00000000	

地址为维护中断控制寄存器（IOR：MT_INT）的地址；中断类型编码为“2b’0x”指示维护中断，为“2’b10”指示睡眠中断，为“2’b11”指示唤醒中断。

5.2.3 维护读存储器命令

字节 0	2’b11	6’b000000/000100（带 CC 标志/不带 CC 标志）
字节 1	地址[7:0]	
字节 2	地址[15:8]	
字节 3	地址[23:16]	
字节 4	地址[31:24]	

字节 5	地址[39:32]
字节 6	8’b00000000
字节 7	8’b00000000

对存储器读命令，读响应的数据长度固定为 128 字节。要求地址必须是 128 字节对界。

5.2.4 维护写存储器命令

字节 0	2’b11	6’b100000/100101/101000/101001 （带 CC 标志字节写/带 CC 标志整块写/ 不带 CC 标志字节写/不带 CC 标志整块写）
字节 1	地址[7:0]	
字节 2	地址[15:8]	
字节 3	地址[23:16]	
字节 4	地址[31:24]	
字节 5	地址[39:32]	
字节 6	数据有效位[7:0]	
字节 7	数据有效位[15:8]	
字节 8	数据 0	
	。。。。。。	
字节 7+n	数据 n-1（n=16 或 128）	

写数据长度只有 128 位或 1024 位两种。当写的长度小于或等于 128 位时，用“字节写”命令，数据有效位表示对应字节的数据有效。如果写数据长度小于 128 字节且大于 128 位时，需要用存储器字节写命令分拆成若干个 128 位来实现。如果写数据长度等于 128 字节，可使用“整块写”命令，此命令的数据有效位无意义。要求地址必须是 16 字节或 128 字节对界。

5.2.5 维护读寄存器命令

字节 0	2’b11	6’b001000
字节 1	地址[7:0]	
字节 2	地址[15:8]	
字节 3	地址[23:16]	
字节 4	地址[31:24]	
字节 5	地址[39:32]	
字节 6	数据有效位[7:0]	
字节 7	2’b00000000	

对一般的 I/O 寄存器，读响应的数据长度固定为 64 位。

PCI-E 接口部件（EP）中的 I/O 寄存器支持 8b、16b、32b 和 64b 四种粒度。四种粒度与低位地

址的对应关系如下：

粒度	数据有效位	地址[2:0]
64b	8’b1111_1111	3’b000
32b	8’b0000_1111	3’b000
	8’b1111_0000	3’b100
16b	8’b0000_0011	3’b000
	8’b0000_1100	3’b010
	8’b0011_0000	3’b100
	8’b1100_0000	3’b110
8b	8’b0000_0001	3’b000
	8’b0000_0010	3’b001
	8’b0000_0100	3’b010
	8’b0000_1000	3’b011
	8’b0001_0000	3’b100
	8’b0010_0000	3’b101
	8’b0100_0000	3’b110
	8’b1000_0000	3’b111

维护接口的读响应数据为 64 位的存储器格式（8b/16b/32b 的响应数据在各自对应位置），维护系统根据请求数据有效位得到相应数据。

5.2.6 维护写寄存器命令

字节 0	2’b11	6’b100100
字节 1	地址[7:0]	
字节 2	地址[15:8]	
字节 3	地址[23:16]	
字节 4	地址[31:24]	
字节 5	地址[39:32]	
字节 6	数据有效位[7:0]	
字节 7	8’b00000000	
字节 8	数据 0	
	。。。。。。	
字节 7+n	数据 n-1(n=8)	

对一般的 I/O 寄存器，写请求的数据长度固定为 64 位。

PCI-E 接口部件（EP）中的 I/O 寄存器支持 8b、16b、32b 和 64b 四种粒度。四种粒度与低位地址的对应关系如下：

粒度	数据有效位	地址[2:0]
64b	8’b1111_1111	3’b000
32b	8’b0000_1111	3’b000
	8’b1111_0000	3’b100
16b	8’b0000_0011	3’b000
	8’b0000_1100	3’b010
	8’b0011_0000	3’b100
	8’b1100_0000	3’b110
8b	8’b0000_0001	3’b000
	8’b0000_0010	3’b001
	8’b0000_0100	3’b010
	8’b0000_1000	3’b011
	8’b0001_0000	3’b100
	8’b0010_0000	3’b101
	8’b0100_0000	3’b110
	8’b1000_0000	3’b111

维护写寄存器的最大数据长度为 8 个字节，最小数据长度为 1 个字节，但命令包中写数据必须为 64 位，且该 64 位数据是存储器格式（8b/16b/32b 的写数据在各自对应位置）。

5.2.7 扫出或监测状态命令

状态监测和状态扫描采用统一编址，共 16MB 的地址空间（具体编址见扫描地址编址说明）。扫出信息统一按 1024 位操作，要求地址[6:0]为全零。

字节 0	2’b11	6’b001100
字节 1	地址[7:0]	
字节 2	地址[15:8]	
字节 3	地址[23:16]	
字节 4	地址[31:24]	
字节 5	地址[39:32]	
字节 6	8’b00000000	
字节 7	8’b00000000	

此命令的响应数据长度固定为 128 字节。

5.2.8 扫入状态命令

状态监测和状态扫描采用统一编址，共 16MB 的地址空间（具体编址见扫描地址编址说明）。扫

入状态分为 64 位和 1024 位两种。1024 位扫入是整块操作，要求地址[6:0]为全零；64 位扫入可以根据数据有效位针对字节操作。

字节 0	2’b11	6’b101101/101111 (字节扫入/整块扫入)
字节 1	地址[7:0]	
字节 2	地址[15:8]	
字节 3	地址[23:16]	
字节 4	地址[31:24]	
字节 5	地址[39:32]	
字节 6	数据有效位[7:0]	
字节 7	数据有效位[15:8]	
字节 8	数据 0	
	。。。。。。	
字节 7+n	数据 n-1（n=16 或 128）	

数据长度只有 128 位和 1024 位两种。当扫入的长度小于或等于 128 位时，用“字节扫入”命令，数据有效位表示对应字节的数据有效。如果扫入数据长度小于 128 字节且大于 128 位时，需要分拆成若干个 128 位来实现。如果扫入数据长度等于 128 字节，可使用“整块扫入”命令，此模块的数据有效位无意义。

5.2.9 维护接口复位命令

执行维护接口复位命令，硬件自动对 6A 内部的维护控制部件的部分逻辑进行复位，便于处理维护命令超时等特殊情况，但对 6A 内部核心运行不产生直接影响。

字节 0	2’b11	6’b110000
字节 1	8’b00000000	
字节 2	8’b00000000	
字节 3	8’b00000000	
字节 4	8’b00000000	
字节 5	8’b00000000	
字节 6	8’b00000000	
字节 7	8’b00000000	

5.2.10 重新存储器自测试命令

重新存储器自测试是通过对 IOR：BIST_GOON 的写来实现。该寄存器只写，写该寄存器将触发 6A 内部重新进行存储器自测试。命令格式中的地址为该寄存器的地址。

字节 0	2’b11	6’b100100
字节 1	地址[7:0]	
字节 2	地址[15:8]	
字节 3	地址[23:16]	
字节 4	地址[31:24]	
字节 5	地址[39:32]	
字节 6	8’b00000000	
字节 7	8’b00000000	
字节 8	8’b00000000	
字节 9	8’b00000000	
字节 10	8’b00000000	
字节 11	8’b00000000	
字节 12	8’b00000000	
字节 13	8’b00000000	
字节 14	8’b00000000	
字节 15	8’b00000000	

5.2.11初始化加载命令

字节 0	2’b11	6’b111100/111111 （初始化加载/初始化加载结束）
字节 1	地址[7:0]	
字节 2	地址[15:8]	
字节 3	地址[23:16]	
字节 4	地址[31:24]	
字节 5	地址[39:32]	
字节 6	8’b00000000	
字节 7	8’b00000000	
字节 8	数据 0	
……	……	
字节 7+n	数据 n-1（n=128）	

初始化加载结束命令不带数据（命令包的长度为 8 字节）。

初始化加载命令的地址（初始化加载结束命令地址没有任何意义）仅作地址检查使用，并不影响申威处理器内部生产的 ICache 加载地址。芯片内部的地址检查仅针对 Srom 加载的“重传”和“漏传”。**芯片默认 Srom 加载地址从 0x0 地址开始，每个 Cache 块加 0x80，且必须按顺序加载。**申威处理器内部实际加载时会根据地址判断该顺序，如果加载地址与期望地址不同（加载地址小于期望地址意味着维护系统发生“重传”，加载地址大于期望地址意味着维护系统发生“漏传”），则发生“重

传”或“漏传”，两种情况仍然返回正常响应（从 5A FPGA 模块的情况看，主要是命令“重传”，**注意不能返回错误响应，否则会反复“重传”**）。申威处理器会对接收到的 Srom 加载命令数量、实际加载 Icache 的命令数量和维护接口复位命令数量进行计数，并设置只读寄存器，共维护和核心读取。

注 1：如果芯片内部检查到进入芯片的维护命令包有错（校验错或格式错等等），则丢弃该包不加载，且加载地址保持不变，并向芯片外部返回错误响应，由外部进行重传。重传的最大次数由芯片外部确定。如果芯片内部正常完成命令包的加载，加载地址递增，并向芯片外部返回正常响应。

注 2：对于判断出的“重传”或“漏传”情况，只要该命令包没有检查到校验错或格式错，都会向芯片外部返回正常响应。

5.2.12Flash 块擦除

字节 0	11	6’b100100 (4K/32K/64K 字节擦除)
字节 1	地址[7:0]	
字节 2	地址[15:8]	
字节 3	地址[23:16]	
字节 4	地址[31:24]	
字节 5	地址[39:32]	
字节 6	00000000	
字节 7	00000000	
字节 8	数据 0	
	。。。。。。	
字节 7+n	数据 n-1(n=8)	

维护命令携带的地址表示 IO 寄存器地址，4K/32K/64K 字节擦除分别对应三个寄存器，写数据对应 4K/32K/64K 字节擦除所对应的 Flash 地址（地址要求与 4.4 节提到的 Flash 读写操作不同，**假设低位对应 Flash 的第 2 个 64 位，即 Flash 域地址 0x8，转换为物理地址其低位也是 0x8**）。写相应的三个寄存器将触发 4K/32K/64K 字节擦除操作。擦除后，该 Flash 块内容为全 1。

5.2.13Flash 全片擦除

字节 0	11	6’b100100
字节 1	地址[7:0]	
字节 2	地址[15:8]	

字节 3	地址[23:16]
字节 4	地址[31:24]
字节 5	地址[39:32]
字节 6	00000000
字节 7	00000000
字节 8	数据 0
	。。。。。。
字节 7+n	数据 n-1(n=8)

维护命令携带的地址表示 I0 寄存器地址，全片擦除对应一个寄存器，写数据没有意义（全片擦除不需要 Flash 地址）。写该寄存器将触发全片擦除操作。擦除后，Flash 内容为全 1。

5.3 维护响应格式：

5.3.1 响应编码

编号	维护响应	编码[7:6]	编码[5:0]	备注
1	带数据读响应（正常读）	固定为 2’b11	6’b000000	带数据（1024 位或 64 位），具体如下： 读存储器：1024 位； 读寄存器是：64 位； 读监测信息或扫出：1024 位。
2	带数据读响应（含有 ECC 多错）		6’b001000	
3	带数据读响应（含有控制错）		6’b010000	
4	非法地址读响应		6’b000001	不带数据
5	写结束		6’b000010	
6	非法地址写结束		6’b000011	
7	带控制错写结束		6’b001011	
8	串行维护接口偶校验错响应		6’b000100	
9	串行维护接口非法维护命令响应		6’b000101	
10	对应接口关闭错误响应		6’b100000	
11	串行处理错误响应		6’b101011	
	保留		其它	

5.3.2 维护响应格式

字节 0	2’b11	响应编码[5:0]
字节 1	数据长度（字节数，只有 0x00、0x08 和 0x80 三种）	
字节 2	数据字节 0	
	……	
字节 1+n	数据字节 n-1(n=0、8 或 128)	

5.3.3 读响应

字节 0	2’b11	6’b000000/001000/010000 （带数据的读响应）
字节 1	数据长度（字节数，只有 0x00、0x08 和 0x80 三种）	
字节 2	数据字节 0	
	……	
字节 1+n	数据字节 n-1（n=8 或 128）	

存储器读的响应数据为 128 字节，寄存器读的响应数据为 64 位（8 字节），状态监测和扫出的响应数据为 1024 位。非法地址读响应没有数据。

5.3.4 其它响应

字节 0	2’b11	响应编码[5:0]
字节 1	8b’00000000	

6 状态扫描地址编址说明

状态监测和扫描统一处理，所有状态监测和扫描信息统一按字节编址，6A 支持 24 位地址，即 16MB 的空间。按模块地址地址空间划分如下：

表 6-1 6A 芯片 SCAN 编址说明

<table>
<tr><th>地址[23:21]</th><th>模块</th><th colspan="2">子模块</th><th>实际可用</th></tr>
<tr><td rowspan="8">3’h0/
3’h1/
3’h2/
3’h3</td><td rowspan="8">CG0/
CG1/
CG2/
CG3</td><td colspan="2">核心 0([20:18]=3’b000)</td><td>8KB，即地址[12:0]</td></tr>
<tr><td colspan="2">核心 1([20:18]=3’b001)</td><td>8KB，即地址[12:0]</td></tr>
<tr><td colspan="2">核心 2([20:18]=3’b010)</td><td>8KB，即地址[12:0]</td></tr>
<tr><td colspan="2">核心 3([20:18]=3’b011)</td><td>8KB，即地址[12:0]</td></tr>
<tr><td rowspan="2">CPM([20:18]
=3’b100)</td><td>CPM([17]=0)</td><td>1KB，即地址[9:0]</td></tr>
<tr><td>TCDATA([17]=1)</td><td>2KB，即地址[10:0]</td></tr>
<tr><td colspan="2">MC0([20:18]=3’b101)</td><td>2KB，即地址[10:0]</td></tr>
<tr><td colspan="2">MC1([20:18]=3’b110)</td><td>2KB，即地址[10:0]</td></tr>
<tr><td colspan="2">保留([20:18]=3’b111)</td><td></td></tr>
<tr><td>3’h4</td><td>CLU</td><td colspan="2"></td><td>32B，即地址[4:0]</td></tr>
<tr><td rowspan="3">3’h5</td><td rowspan="3">ION&
INTPU</td><td colspan="2">INTPU([20:19]=2’b0x)</td><td>1KB，即地址[9:0]</td></tr>
<tr><td colspan="2">ION0([20:19]=2’b10)</td><td>1KB，即地址[9:0]</td></tr>
<tr><td colspan="2">ION1([20:19]=2’b11)</td><td>1KB，即地址[9:0]</td></tr>
<tr><td rowspan="2">3’h6</td><td rowspan="2">PIU0</td><td colspan="2">PIU0([20:18]=3’bxx0)</td><td>1KB，即地址[9:0]</td></tr>
<tr><td colspan="2">PageCache0([20:18]=3’bxx1)</td><td>256B，即地址[7:0]</td></tr>
<tr><td rowspan="3">3’h7</td><td rowspan="3">PIU1&MCU</td><td colspan="2">PIU1([20:18]=3’b0x0)</td><td>1KB，即地址[9:0]</td></tr>
<tr><td colspan="2">PageCache1([20:18]=3’b0x1)</td><td>256B，即地址[7:0]</td></tr>
<tr><td colspan="2">MCU([20:19]=2’b10)</td><td>1KB，即地址[9:0]</td></tr>
<tr><td>其它</td><td>—</td><td colspan="2">—</td><td>保留</td></tr>
</table>

7 基于 JTAG 的维护操作

通过 JTAG 端口可访问的维护接口寄存器，实现对芯片的维护操作。图 7-1 给出了基于 JTAG 接口的维护控制的结构图，主要包括维护接口寄存器和维护处理逻辑。

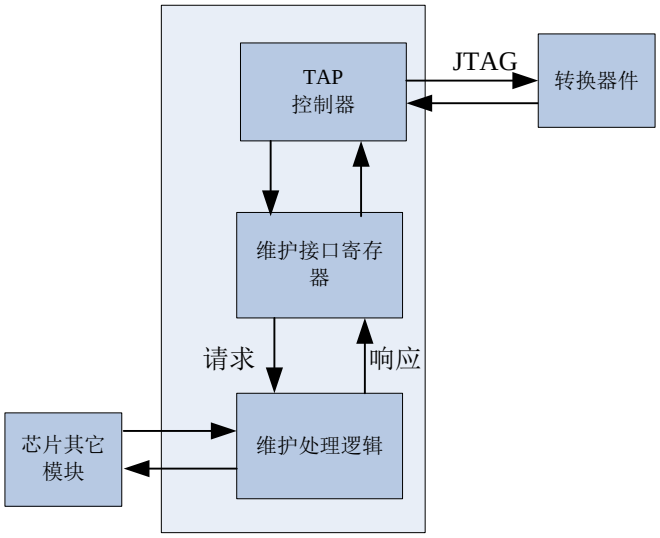

图 7-1：JTAG 接口结构

注：TAP 控制相关信息见《6A 测试接口协议》。

7.1 JTAG 可访问的维护接口寄存器

7.1.1 接口寄存器

JTAG（TAP 控制器）通过接口寄存器与 MCU 进行交互。

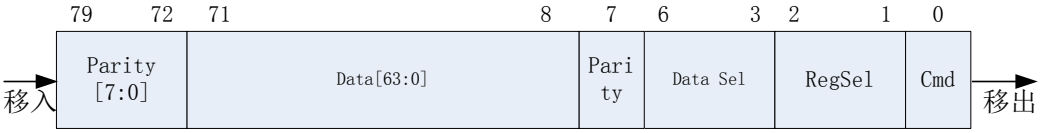

图 7-2：MCU 接口寄存器

1) 位宽：80bit。

   a) Cmd 表示操作类型，0 表示读，1 表示写；读写相对此次操作对应的寄存器（RegSel[1:0]选择）。

   b) RegSel[1:0]表示寄存器选择，0 表示命令寄存器，1 表示数据寄存器。2 或 3 表示状态寄存器；

   c) DataSel[3:0]表示数据 Wrap 选择，只对写数据或读响应数据有意义，表示本次数据 Wrap 地址（按 64 位而言）。

d) Data[63:0]表示本次读写操作（Cmd）的内容，如果操作命令寄存器则该数据为命令本身。

e) Parity 表示偶校验位。其中 bit7 是 bit[6:0]的偶校验位，bit[79:72]是 bit[71:8]的偶校验位，每个校验位对应 8 位数据。

2) 串行扫入格式要求：按照字节 1（低位到高位） +字节 2（低位到高位）+……+字节 10（低位到高位）的顺序进行。

3) 串行扫出格式要求：按照字节 1（低位到高位） +字节 2（低位到高位）+……+字节 10（低位到高位）的顺序进行。

7.1.2 命令寄存器

对该寄存器执行 Capture 操作（Tap 控制器动作），则返回全“1”的特征数据。

对接口寄存器进行 Update 时（Tap 控制器动作），如果 Cmd 为“1”（表示写操作）、RegSel[1:0]为命令寄存器， Data[63:0]为携带的命令。此时，首先把 Update 动作异步交接到维护时钟域，并触发状态机切换。该维护命令被直接发送到 RTPU 模块（MCU 内部子模块），如果 RTPU 模块正在处理上一个维护请求，则直接返回错误响应。

7.1.3 数据寄存器

对接口寄存器进行 Update 时（Tap 控制器动作），如果 Cmd 为“1”、RegSel[1:0]为数据寄存器，Data[63:0]为携带的写数据、DataSel[3:0]是本次 64 位数据所对应的 Wrap 地址（要求系统 Wrap 地址从 0 开始）。此时，首先把 Update 动作异步交接到维护时钟域，并触发状态机切换。该维护写数据被直接发送到 RTPU 模块，全部写数据收齐后进行下一步操作。

对接口寄存器进行 Update 时（Tap 控制器动作），如果 Cmd 为“0”、RegSel[1:0]为数据寄存器，Data[63:0]无意义，DataSel[3:0]是读响应数据所对应的 Wrap 地址。此时，首先把 Update 动作异步交接到维护时钟域，根据 Wrap 地址得到 64 位响应数据写接口寄存器 Data[63:0]（需异步交接）。然后在对接口寄存器进行 Capture（Tap 控制器动作），得到该响应数据。

7.1.4状态寄存器

该寄存器中部分域（STATE、RES_ERR）的寄存器设置在 TBXI 模块，其它位信息直接来自 MCU 的其它子模块。维护时钟域和 TCLK 的异步问题可以通过对状态寄存器多次读保证。**该寄存器只读。**

表 7-1：TBXI 模块状态寄存器

描述符	域	意义

RSV	[63:38]	保留。
OBS	[37:33]	观测输出（仅选择能稳定输出的信号）。 [37]:保留。 [36]:保留。 [35]:保留。 [34]: PCIe-1 接收方链路连接建立。 [33]: PCIe-0 接收方链路连接建立。
FLAG	[32:28]	[32]：保留； [31:28]：对应 IOR：FlagReg[3:0]。该寄存器核心可读写，通过编码表示各类状态，具体状态信息意义由软件控制。
BIST_INF	[27:24]	[27]：BIST 测试完成，**高电平有效。** [26:25]：测试结果。 2’b00，表示测试无错； 2’b01，表示有错可修复； 2’b10，表示有错不可修复； 2’b11，表示测试时间超时。 [24]：指示在存储器 Debug 测试方式下，发现错误而暂停存储器 Debug 测试，此时可通过状态扫描获得具体的错误信息，**高电平有效。**
RES_ERR	[23]	JTAG 接口维护控制错，**高电平有效。**
SYS_ERR	[22]	系统错标志，**高电平有效**。
INT_DONE	[21]	中断完成标志，**高电平有效**。
RSV	[20]	保留
MFSM	[19:16]	维护主状态机编码。
RES_LNT	[15:8]	响应长度。 0：0 字节； 8：8 字节； 128：128 字节；

		其它保留。
RES_CMD	[7:2]	响应编码[5:0]。
STATE	[1:0]	记录基于 JTAG 接口的维护请求处理状态。 2’b00：空闲状态。 2’b01：维护请求发送状态。 2’b10：维护请求处理状态。 2’b11：维护命令处理结束状态。 对于维护命令（扫入维护命令寄存器）必须查询该状态标志，等待该标志为空闲状态。

对接口寄存器进行 Update 时（Tap 控制器动作），如果 Cmd 为“0”、RegSel[1:0]为状态寄存器，Data[63:0]无意义，DataSel[3:0]无意义（只有一个 64 位，默认为零）。此时，首先把 Update 动作异步交接到维护时钟域，得到 64 位响应数据写接口寄存器 Data[63:0]（需异步交接）。然后在对接口寄存器进行 Capture，得到该响应数据。

对状态寄存器的写只用于触发状态机切换。

7.2 操作流程

本节主要说明基于标准 JTAG 的维护操作流程，该过程可以作为底层驱动（结合《6A 测试接口协议》中关于 Tap 控制器的内容）。

1)MCU 链选择

a)对 TBOX 指令寄存器进行 Update，写入 CHAINADDR 命令；

b)对 TBOX 数据寄存器进行 Update，由于当前指令寄存器是 CHAINADDR 命令，所以此移入的数据是 ScanAddrReg，选择 MCU 接口寄存器；

c)对 TBOX 指令寄存器进行 Update，写入 SAMPLE/PRELOD 或 EXTEST 命令；

d)此时可以操作 MCU 的链。

2)MCU 接口状态寄存器获取。

a)对 MCU 接口寄存器进行 Update，选择状态寄存器并进行读操作（Cmd 为零，RegSel[1:0]选择数据寄存器）；

b)中间等待 16 个 TCLK，用于异步交接（大于 TCLK->MCLK->TCLK 两次异步交接的延时），以保证接口寄存器内容稳定。

c)对 MCU 接口寄存器进行 Capture，得到 MCU 接口状态寄存器，如果允许发送命令（表

7-1 的 STATE，下同）则发送命令，否则反复查询。

3)维护读命令操作流程

a)对 MCU 接口寄存器进行 Update，选择读命令、读地址（Cmd 为“1”，RegSel[1:0]选择命令寄存器，Data[63:0]用于指定维护请求类型，具体见第五章）；

b)MCU 进行异步交接后触发维护操作，如果此时有其它接口的维护命令正在运行，则生成错误响应；

c)对接口寄存器进行 Update，读取状态寄存器内容（详见步骤 2），如果响应已经准备好且有响应数据，则准备得到本次读响应数据（子步骤 d~f）；

d)对接口寄存器进行 Update，选择数据寄存器并进行读操作（Cmd 为“0”，RegSel[1:0]选择数据寄存器）；

e)中间等待 16 个 TCLK，用于异步交接（大于 TCLK->MCLK->TCLK 两次异步交接的延时）。

f)对接口寄存器进行 Capture，得到数据寄存器中响应数据；**如果响应数据为 1024 位，则反复步骤 3 中的 d~f 子步骤。**

g)**上层软件根据维护响应长度指示得到全部的响应数据后，对接口寄存器进行 Update，选择状态寄存器并进行写操作（Cmd 为“1”，RegSel[1:0]选择数据寄存器），该动作仅仅用于交互。**

4)维护写命令操作流程

a)对接口寄存器进行 Update，选择写命令、写地址（Cmd 为“1”，RegSel[1:0]选择命令寄存器，Data[63:0]用于指定维护请求类型，具体见第五章）；

b)MCU 进行异步交接后触发维护操作，如果此时有其它接口的维护命令正在运行，则生成错误响应，否则等待后续写数据后启动维护操作；

c)对接口寄存器进行 Update，选择写数据（Cmd 为“1”，RegSel[1:0]选择数据寄存器，Data[63:0]为写数据）；**如果写数据为 128 位或 1024 位，则反复该子步骤。上层软件根据维护请求数据长度指示移入写数据，全部数据结束后，开始维护操作。**

d)对接口寄存器进行 Update，读取状态寄存器内容（详见步骤 2），如果响应已经准备好（不携带数据），则已经得到响应；

e)**对接口寄存器进行 Update，选择状态寄存器并进行写操作（Cmd 为“1”，RegSel[1:0]选择数据寄存器），该动作仅仅用于交互。**

5)维护操作注意事项：

a)系统通过 JTAG 端口读状态寄存器 STAT_REG[19:16]可确定维护主状态机所处的状态。系统可在配置 1 状态、配置 2 状态、配置 3 状态、初始化加载状态、运行状态下发送维护命令；

b)当要发送维护命令时，系统通过 JTAG 端口读状态寄存器 STAT_REG[1:0]，如果此时 JTAG 接口的维护请求处理状态处在空闲状态，则系统通过 JTAG 端口发送维护命令；

c)系统循环查询状态寄存器，当确定状态寄存器 STAT_REG[1:0]= 2’b11 时，说明此时响应已经收齐，根据 STAT_REG[15:2]判断命令类型和响应数据长度。**无论是否有读响应数据，系统都需要写状态寄存器，通知状态机切换到空闲状态。**。

说明：如果维护请求超时，系统可以发送“维护接口复位命令”。“维护接口复位命令”可以复位维护请求处理模块的相关逻辑。**“维护接口复位命令”不需要判断 JTAG 端口状态寄存器 STAT_REG[1:0]，任何状态“维护接口复位命令”总能够被正确识别。**

如果维护请求处理模块正在处理维护请求，且该请求是由于 JTAG 接口发起，此时又收到来自 JTAG 接口的新请求，则作为维护接口错（见状态寄存器）。如果响应数据发送期间，收到对数据寄存器的 Update，则作为维护接口错（见状态寄存器）。该错误在收到下一个基于 JTAG 接口的维护请求（对命令寄存器的写）或“维护接口复位命令”时被清除。

7.3 芯片冷复位

针对基于 JTAG 的维护操作增加特殊的维护命令用于冷复位。**该命令仅对 JTAG 接口的维护操作开放，**执行该命令则形成一个 1000 个维护时钟周期的复位脉冲（低有效）。

表 7-1：冷复位命令

字节 0	2’b11	6’b110011
字节 1	8’b00000000	
字节 2	8’b00000000	
字节 3	8’b00000000	
字节 4	8’b00000000	
字节 5	8’b00000000	
字节 6	8’b00000000	
字节 7	8’b00000000	