玄铁 C910 集成手册（openc910）

2021 年 10 月 19 日

版本	描述	日期
01	openc910 第一版发布。	2021.10.19

玄铁 C910 集成手册

第一章　概述

1.1　处理器简介

玄铁 C910 是面向嵌入式系统和 SoC 应用领域的 64 位超高性能嵌入式多处理器核，具有出色的性能表现。C910 采用了 RISCV 64GC 基本指令集，和平头哥性能增强指令集，主要面向对性能要求严格的高端嵌入式应用，如人工智能、自动驾驶、移动智能终端、高性能通信、信息安全等等。

C910MP 采用同构多核架构，支持 2 个 C910 核心。每个 C910 核心采用自主设计的体系结构和微体系结构，并重点针对性能进行优化，引入 3 发射 8 执行的超标量架构、多通道的数据预取等高性能技术。C910 支持实时检测并关断内部空闲功能模块，进一步降低处理器动态功耗。

C910 的微体系结构图如 图 1.1 所示。

C910 处理器体系结构的主要特点如下：

- 同构多核架构，支持 2 个 C910 核心可配置；
- 支持 AXI4.0 Master 接口，128 比特的总线宽度；
- 两级高缓结构，哈佛结构一级高缓和共享的二级高缓；
- 一级缓存支持 MESI 的一致性协议，二级缓存支持 MOESI 的一致性协议；
- 二级高缓支持 16 路组相联；
- 二级高缓大小为 1MB，缓存行 SIZE 为 64B；
- 支持私有中断控制器 CLINT 和公有中断控制器 PLIC；
- 支持计时器功能；
- 支持自定义的多核调试框架。

每个 C910 核心的主要特征点如下：

- RISC-V 64GC 指令架构；
- 支持小端模式；
- 9~12 级深流水架构；
- 3 发射 8 执行的超标量架构，对软件完全透明；

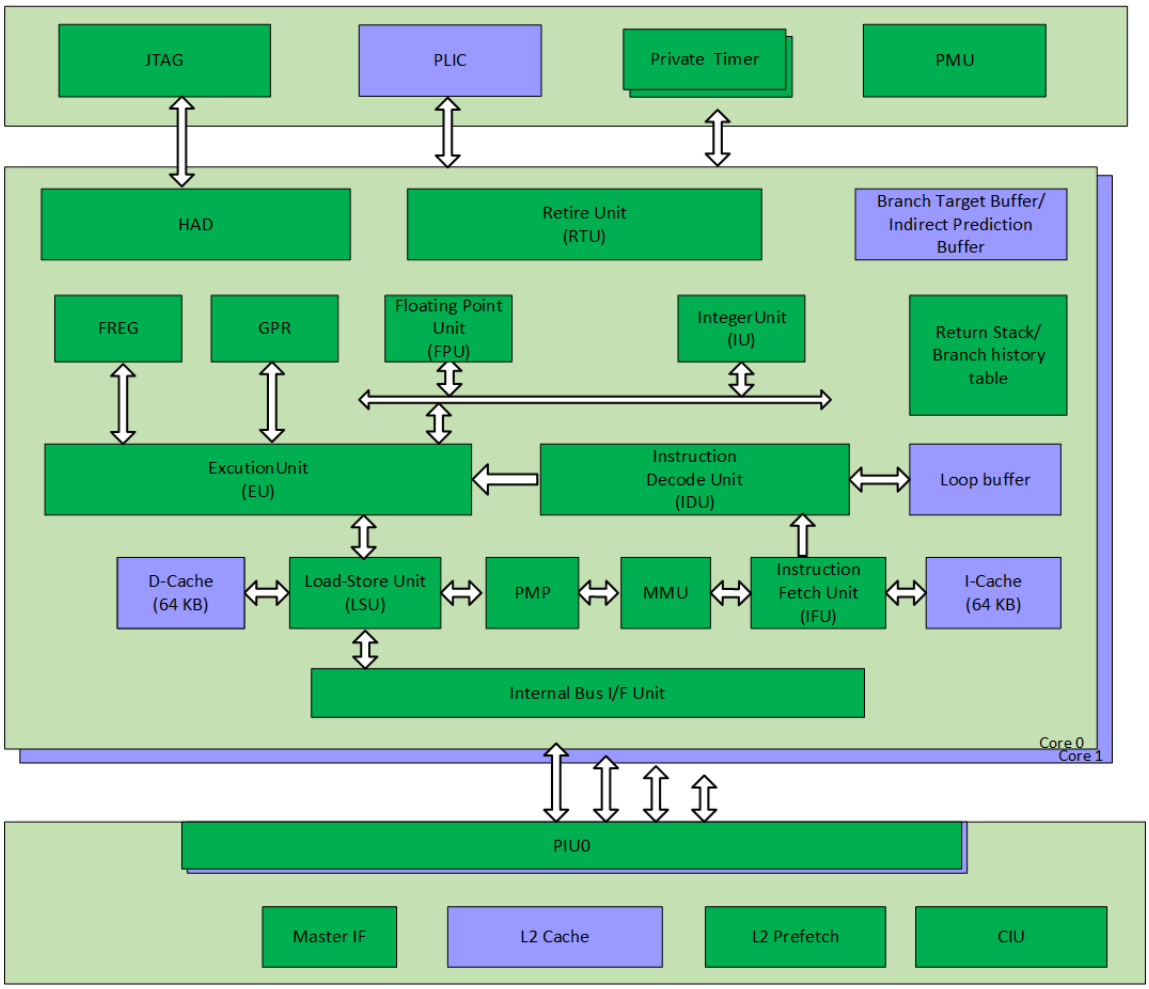


图 1.1: C910 微体系结构图

- 按序取指，乱序发射，乱序完成和按序退休；
- 两级 TLB 内存管理单元，实现虚实地址转换与内存管理；
- 指令高缓和数据高缓大小为 64KB，缓存行为 64B；
- 指令预取功能，硬件自动检测并动态启动；
- 指令高缓路预测的低功耗访问技术；
- 短循环缓存的低功耗执行技术；
- 64Kb 的两级多路并行分支预测器；
- 支持 12 层的硬件返回地址堆栈；
- 256 表项的间接跳转分支预测器；
- 非阻塞发射，投机猜测执行；
- 基于物理寄存器的重命名技术；
- 支持 0 延时 move 指令；
- 双发射、全乱序执行的 load、store 指令；
- 支持读写各 8 路并发的总线访问；
- 支持写合并；
- 支持 8 个通道的数据缓存硬件预取，支持 stride 的预取方式；
- 浮点执行单元可配置，支持半精度、单精度和双精度；

C910 单核心主要的工艺和性能参数如下：

- TSMC12nm 工艺下，工作频率高于 2GHz（典型应用场景）；
- 性能 6.0 DMIPS/MHz，7.0 coremark/MHz。

第二章　玄铁 C910 RTL 的配置

2.1　玄铁 C910 IP 数据包介绍

玄铁 C910 IP 包由以下两部分组成：C910 源码包，包含固定功能配置的 C910 RTL 代码；C910 Smart 平台，提供了 C910 的参考集成设计、仿真环境和测试用例等，帮助熟悉 C910 的功能和使用方法，并辅助 C910 的集成工作。功能包括：

- C910 的参考集成设计；
- 丰富的测试用例：1）与 C910 相关的 firmware 参考代码，包括 C910 的最佳性能设置、中断/异常处理，低功耗处理流程等；2）帮助客户对 C910 的集成的正确性做检查；
- 基础的软硬件仿真环境；
- 后端综合实现的参考 SDC 约束文件；
- 完整的 C910 低功耗策略演示（包含 UPF）；
- 可移植 RV Platform Wrapper。

2.2　玄铁 C910 IP 包的环境要求

在使用玄铁 C910 IP 包之前，请检查并确保运行环境备以下条件：

- 玄铁 C910 IP 中的 perl 脚本依赖 perl 5.10.1 或以上版本；
- Make 版本为 3.8.1；
- EDA 仿真工具：Icarus Verilog (iverilog) 10.2，Synopsys VCS 2019.06 及以上版本，或者 Cadence Xcelium 19.09 及以上版本；带 UPF 的低功耗仿真仅在 VCS 2020.12 版本进行了测试。
- Smart 平台编译测试用例依赖的平头哥玄铁处理器 RISC-V 工具链 2.0.3 及以上版本。

2.3　C910 配置说明

C910 功能配置说明如 表 2.1 所示。

表 2.1: C910 功能配置

可配置单元	配置选项	详细
Core Num	2	openc910 提供了 2 个 C910 核心的可配。
L1 DCache	64KB	openc910 提供了 64KB 的 L1DCACHE。
L1 Icache	64KB	openc910 提供了 64KB 的 L1ICACHE。
L2 CACHE	1MB	openc910 提供了 1MB 大小的 L2CACHE 配置。
中断数量	16-1008 个	openc910 提供了 144 个外部中断接入。

2.4　Memory 替换

2.4.1　需作替换处理文件所在位置

所有需要修改的文件都在 “INST_LIB” 目录，且文件名都包含关键词 “ct_*_spsram_AAAxBBB”，A、B 分别代表 memory 的深度和宽度。

2.4.2　Memory 规格

所有需要的 memory 的深度以及宽度规格可以从 INST_LIB 下文件的 AAAxBBB 命名中得到。

另外，我们对所需的 memory 有一个要求，需要支持是位写使能信号。其他的选项用户可以根据自己的需求选择，包括 memory 的形状、时序、面积、功耗等因素。如果觉得有些 memory 过大或者过深，可以自行进行拼接，后面章节会介绍 memory 的拼接。

2.4.3　Memory 端口说明

以特定工艺 memory 为例，如 图 2.1 所示。

```
sprf065ll_512x64 x_spsram_512x64(
  .A    (A),
  .D    (D),
  .BWEN ({{32{WEN[1]}}, {32{WEN[0]}}}),
  .WEN  (&WEN),
  .CEN  (CEN),
  .CLK  (CLK),
  .Q    (Q)
);
```

图 2.1: Memory 例化

Memory port 信号列表如 表 2.2 所示。

表 2.2: memory port 信号列表

信号线名	功能	连接
A	地址线。	RAM 的地址线端口。
D	写入数据线。	RAM 的数据输入端。
WEN*	低电平有效，位写使能信号。	RAM 位写使能信号端缩位与之后做 RAM 写使能信号。
CEN*	低电平有效，位片选信号。	RAM 使能信号。
CLK	时钟。	RAM 时钟端。
Q	输出数据线。	RAM 的输出。

这两根信号都是低电平有效，具体揭发要看 RAM 的相应信号是否也是低电平有效，另外，不同 Vender 提供 RAM 可能还有一些其他的控制 port，针对这些 Port 用户需要根据自己的需要按照 RAM Vender 的用户手册进行处理。

2.4.4 Memory 控制与行为描述

图 2.2 是一个 RAM 的读写时序图，Trans A 是一个写请求，Trans B 是一个读请求。写部分数据是 BWEN 控制的。Trans B 时钟上升沿采到到输入读请求，下一个 Cycle 将数据输出到 Q 端。

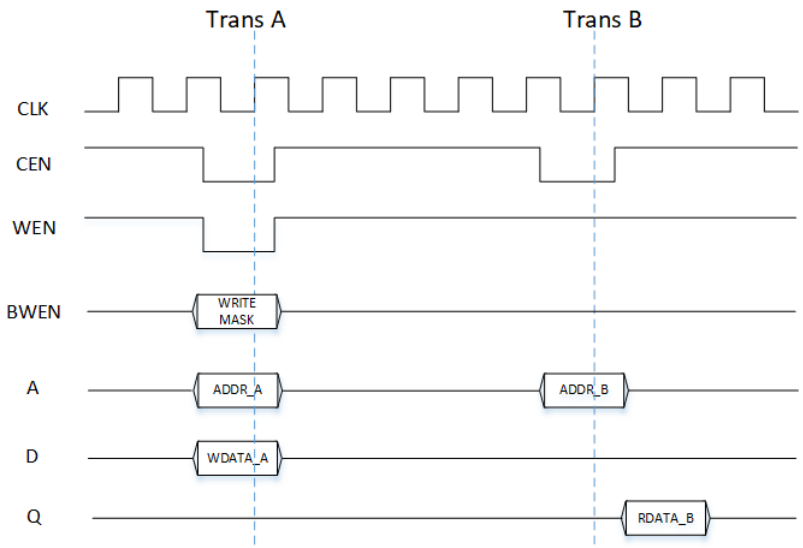


图 2.2: Memory 读写时序

2.4.5 拼接 Memory

如所需 memory 规格在特定 compiler 下无法生成，可以用更小的 memory 拼接出需要的 memory。

以 2048x32 的 memory 为例如 图 2.3 所示。

```
spsram040g_2048x32 x_spsram_2048x32(
  .A    (A),
  .D    (D),
  .BWEB ({{8{WEN[3]}},{8{WEN[2]}},{8{WEN[1]}},{8{WEN[0]}}}),
  .WEB  (&WEN),
  .CEB  (CEN),
  .CLK  (CLK),
  .DELAY(2'b00),
  .TEST (2'b00),
  .Q    (Q)
);
```

图 2.3: Memory 拼接示例

假设某个工艺下并不支持该规格的 memory，就需要用用其他规格 memory 进行拼接，用两块 1024x32 的 memory 进行拼接的示例如 图 2.4 所示。

2.5　ICG 替换

2.5.1　需作替换处理文件所在位置

所有需要修改的文件在 INST_LIB 目录，文件名位 “gated_clk_cell.v”，该文件里例化了特定工艺下的 Gated cell。

2.5.2　ICG 选用

玄铁 CPU 所用的 ICG 单元都是上升沿有效的，其逻辑图和真值表如 图 2.5 所示。

一般工艺库都会有多类 ICG，一类是正沿触发而另一类是负沿触发的。需注意选择和以上真值表一样的正沿 ICG。

以特定工艺 ICG 单元为例如 图 2.6 所示。

2.5.3　ICG 旁路

在特定情况希望旁路 RTL 中的 ICG 单元功能时（比如用作 FPGA 制作），可以使用 图 2.7 改法处理。

2.6　Smart 平台介绍

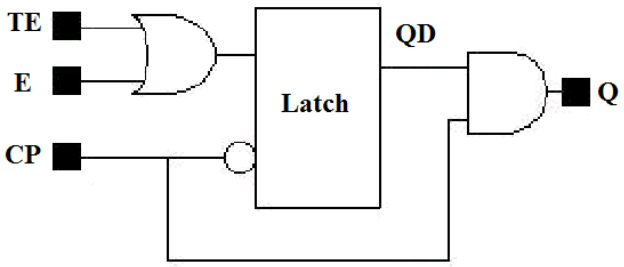
C910 配套提供了简要的 SoC 集成仿真参考环境：Smart。

```
assign CEN0 = CEN |  A[ADDR_WIDTH-1];
assign CEN1 = CEN | ~A[ADDR_WIDTH-1];

always@(posedge CLK)
begin
  if (!CEN)
  begin
    bank_sel <= A[ADDR_WIDTH-1];
  end
  else
  begin
    bank_sel <= bank_sel;
  end
end
assign  Q[31:0] = bank_sel ? Q1[31:0] : Q0[31:0];
sprf065lp_1024x32 x_spsram_1024x32_bank0(
      .A    (A[ADDR_WIDTH-2:0]),
      .D    (D),
      .BWEB ({ {8{WEN[3]}}, {8{WEN[2]}},{8{WEN[1]}}, {8{WEN[0]}} }
            ),
      .WEB  (&WEN),
      .CEB  (CEN0),
      .CLK  (CLK),
      .TURBO(1'b1),
      .RTSEL(1'b0),
      .TSEL (2'b01),
      .Q    (Q0)
    );

  sprf065lp_1024x32 x_spsram_1024x32_bank1(
      .A    (A[ADDR_WIDTH-2:0]),
      .D    (D),
      .BWEB ({ {8{WEN[3]}}, {8{WEN[2]}},{8{WEN[1]}}, {8{WEN[0]}} }
            ),
      .WEB  (&WEN),
      .CEB  (CEN1),
      .CLK  (CLK),
      .TURBO(1'b1),
      .RTSEL(1'b0),
      .TSEL (2'b01),
      .Q    (Q1)
    );
```

图 2.4: Memory 拼接示例



Truth Table

INPUT			OUTPUT
TE	CP	E	Q
0	↑	0	0
0	↑	1	1
1	↑	x	1
x	0	x	0
x	1	x	QD

图 2.5: 正沿触发 ICG

```
CKLNQD8BWP30P140 x_gated_clk_cell (
                 .CP             (clk_in          ),
                 .TE             (SE              ),
                 .E              (clk_en_bf_latch),
                 .Q              (clk_out         )
                 );
```

图 2.6: ICG 的例化

```
//    CKLNQD8BWP30P140 x_gated_clk_cell (
//                     .CP           (clk_in),
//                     .TE             (SE),
//                     .E      (clk_en_bf_latch),
//                     .Q          (clk_out)
//                     );

assign clk_out = clk_in;
```

图 2.7: ICG 旁路示例

2.6.1 Smart 平台的使用

测试用例

执行如下命令，可以列出所有当前 C910 配置下，Smart 提供的测试用例:

```
make showcase
```

在仿真中执行每个测试用例前，需要做测试用例的编译，命令为:

```
make buildcase CASE= 测试用例名（测试用例名可从上述 make showcase 命令的执行结果中挑选）
```

需要注意，Smart 平台依赖环境变量 $TOOL_EXTENSION 来选择编译测试用的 RISC-V 编译器，Cmart 平台在目录 C910_SMART_FACTORY/setup/下，提供了名为 example_setup.csh 的脚本作为参考，举例如何设置 $TOOL_EXTENSION。

Smart 平台 Testbench 编译

在仿真前，需要做硬件编译，包括：1）Smart 平台提供的 C910 的参考集成设计 2）Smart 平台的仿真环境

命令如下:: make compile

测试用例的执行

如果 Testbench 编译和测试用例编译都已经完成，执行测试用例的命令为:

```
make runcase CASE=测试用例名 -run
```

可以用以下命令，直接 Testbhench 编译、测试用例编译与单个测试用例的执行:

```
make runcase CASE=测试用例名
```

执行 Smart 平台提供的全部测试用例的命令为:

```
make regress
```

当所有用例执行结束后，命令行中将显示包含所有测试用例执行结果的报告，该报告也可以在…/smart_run/tests/regress 路径下找到。

测试用例的调试

当测试用例的执行有异常时，可以在 Smart 平台里生成波形帮助问题定位，命令为:

```
make runcase CASE=测试用例名 DUMP=on
```

UPF 测试用例的执行

C910 Smart 平台提供了 UPF 的参考设计，并提供了带 UPF 的低功耗仿真环境，执行 UPF 低功耗方正的命令为:

```
make runcase CASE=sleep DUMP=on UPF=on
```

需要注意的是，目前 C910 Smart 平台的低功耗仿真仅支持仿真器 VCS 2020.12 及以上版本。

请执行以下命令获得更多关于 Smart 环境使用的信息:

```
make help
```

第三章　集成总览

本节主要介绍 C910 的接口集成概要，为了降低 C910 集成用户的硬件集成复杂度，C910 除了总线接口部分的信号外，其他信号在玄铁多款系列处理器中，均保持了稳定的延续性，以方便 C910 的升级换代。根据 C910 顶层端口信号的特点，划分为时钟复位信号，总线系统信号，中断系统信号，调试系统信号，低功耗系统信号，DFT 系统信号，CPU 运行观测信号等几大类。

3.1　命名规则

表 3.1: 信号命名规则

信号名前缀	描述
pad_biu_*	输入信号。
pad_had_*	
pad_cpu_*	
pad_corex_*	
biu_pad_*	输出信号。
had_pad_*	
cpu_pad_*	
corex_pad_*	
*_b	低电平有效信号。

一般来说，没有特殊规定，C910 的输入输出信号均为高电平有效，但是注意，如果是以“_b”结尾的信号，则是低电平有效。

3.2　端口信号列表

端口信号列表如 表 3.2 所示。

表 3.2: 端口信号列表

信号名	I/O	初始值	时钟域	功能描述
时钟复位信号集成				
状态和时钟信号				
pll_cpu_clk	I	-	-	全局工作时钟信号: 提供各个处理器核和 L2Ca che 的工作时钟。
axim_clk_en	I	-	CPU	主设备接口与总线同步时钟使能信号: 主设备接口与总线同步时钟使能有效。
复位信号				
pad_cpu_rst_b	I	-	异步	全局复位信号: 低电平时，初始化 C92 0MP 除 C910 核心外的模块，包括 L2 cache、中断控制器、mast er 接口和 slave 接口。C910 采用异步复位方式。
pad_core(x)_rvba[39:0]	I		同步	core(x) 的复位启动地址信号: pad_core(x)_rvba[0] 需为零。该信号需为常值或者同步到 cpu 时钟域。
pad_core(x)_rst_b	I	-	异步	各个处理器核复位信号: 低电平时，初始化 C 910 核心、及其调试辅助模块和私有计时器，采用异步复位方式。
pad_had_jtg_trst_b	I	-	JTAG	JTA G 测试复位信号: JTA G 复位信号，下跳变可以初始化和 JTAG 接口相关的触发器,正常工作情况下该信号需置成高电平。
总线系统集成				
AXI 总线主接口信号				
biu_pad_araddr[39:0]	O	40’b0	SYS	读地址总线: 40 位地址总线。
biu_pad_arburst[1:0]	O	2’b0	SYS	突发传输指示信号: 指示传输是一次突发传输的一部分: 01：INCR; 10：WRAP4。
biu_pad_arcache[3:0]	O	4’b0	SYS	读请求对应的 cache 属性: [3]:Other Allocate; [2]:Allocate; [1]:Modifiable; [0]:Bufferable。
biu_pad_arid[7:0]	O	8’b0	SYS	读地址 ID。

下页继续

表 3.2 – 续上页

信号名	I/O	初始值	时钟域	功能描述
biu_pad_arlen[7:0]	O	8’b0	SYS	突发传输长度： 00000000：1 拍; 00000011：4 拍。
biu_pad_arlock	O	1’b0	SYS	读请求对应的访问方式： 0：normal access； 1：exclusive access。
biu_pad_arprot[2:0]	O	3’b0	SYS	读请求的保护类型： 0 | 1 [2]: Data | Instruction； [1]: Secure | Non-secure； [0]: User | Privileged。
biu_pad_arsize[2:0]	O	3’b0	SYS	读请求每拍数据位宽： 000: 8bits; 001:16bits; 010:32bits; 011:64bits; 100: 128bits。
biu_pad_arvalid	O	1’b0	SYS	读地址有效信号。
pad_biu_arready	I	-	SYS	读地址通道 ready 信号。
pad_biu_rdata[127:0]	I	-	SYS	读数据总线： 128 位数据总线。
pad_biu_rid[7:0]	I	-	SYS	读数据 ID。
pad_biu_rresp[3:0]	I	-	SYS	读响应信号： [1:0]： 00：OKAY； 01：EXOKAY； 10：SLVERR； 11：DECERR。
pad_biu_rlast	I	-	SYS	读数据最后一拍指示信号。
pad_biu_rvalid	I	-	SYS	读数据有效信号。
biu_pad_rready	O	1’b1	SYS	读数据通道 ready 信号。
biu_p ad_awaddr[39:0]	O	40’b0	SYS	写地址总线： 40 位地址总线。
biu_pad_awburst[1:0]	O	2’b0	SYS	突发传输指示信号： 指示传输是一次突发传输的一部分： 01：INCR；

下页继续

表 3.2 – 续上页

信号名	I/O	初始值	时钟域	功能描述
biu_pad_awcache[3:0]	O	4’b0	SYS	写请求对应的 cache 属性： [3]:Other Allocate； [2]:Allocate； [1]:Modifiable； [0]:Bufferable。
biu_pad_awid[7:0]	O	8’b0	SYS	写地址 ID。
biu_pad_awlen[7:0]	O	8’b0	SYS	突发传输长度： 00000000：1 拍； 00000011：4 拍。
biu_pad_awlock	O	1’b0	SYS	写请求对应的访问方式： 0：normal access； 1：exclusive access。
biu_pad_awprot[2:0]	O	3’b0	SYS	写请求的保护类型： 0 | 1 [2]: Data | Instruction； [1]: Secure | Non-secure； [0]: User | Privileged。
biu_pad_awsize[2:0]	O	3’b0	SYS	写请求每拍数据位宽： 000: 8bits； 001:16bits； 010:32bits； 011:64bits； 100: 128bits。
biu_pad_awvalid	O	1’b0	SYS	写地址有效信号。
pad_biu_awready	I	-	SYS	写地址通道 ready 信号。
biu_pad_wvalid	O	1’b0	SYS	写数据有效信号。
pad_biu_wready	I	-	SYS	写数据通道 ready 信号。
pad_biu_bid[7:0]	I	-	SYS	写响应 ID。
pad_biu_bresp[1:0]	I	-	SYS	写响应信号： 00：OKAY； 01：EXOKAY； 10：SLVERR； 11：DECERR。
pad_biu_bvalid	I	-	SYS	写响应有效信号。
biu_pad_bready	O	1’b1	SYS	写响应通道 ready 信号。
biu_pad_cactive	O	1’b1	-	固定为 1。
biu_pad_csysack	O	1’b1	SYS	低功耗通道请求响应信号。
pad_biu_csysreq	I	-	SYS	低功耗通道请求信号。

下页继续

表 3.2 – 续上页

信号名	I/O	初始值	时钟域	功能描述
中断系统集成				
pad_cpu _apb_base[39:0]	I	-	SYS	片上中断控制器等基地址信号。该信号低 27 为需为零。
pad_plic _int_vld[i-1:0]	I		异步	公有中断源的中断指示信号: 表示公有中断源的中断是否有效: 1：表示中断有效; 0：表示中断无效。 注: i 表示中断源的数量。该信号在 C910 内部同步到 cpu 时钟域。
p ad_plic_int_cfg [i-1:0]	I		异步	公有中断源的中断配置信号: 表示公有中断源的中断属性: 1：表示脉冲中断; 0：表示电平中断。 该信号需接常值或者先同步到 cpu 时钟域。
pad_cp u_sys_cnt[63:0]	I		SYS	外部系统时钟，该信号需要由外部同步到 SYS 时钟域。
调试系统集成				
JTAG 支持信号				
p ad_had_jtg_tclk	I	-	-	JTA G 测试时钟信号: JTAG 和 HAD 内部相关触发器的时钟信号，要求和 CPU 时钟的频率比在 1：8 以上。
pad_had_jtg_tms	I	-	JTAG	JTAG 5 控制输入信号: JTAG 模式选择信号，控制 HAD 中 TAP 状态机的运作
pad_had_jtg_tdi	I	-	JTAG	JTAG 5 数据输入信号: JTAG 5 串行输入端口，包括控制命令，数据，输入顺序由低位到高位。
had_pad_jtg_tdo	O	-	JTAG	JTAG 5 数据输出信号: JT AG5 串行数据输出端口，输出顺序由低位到高位。
had_pad_jtg_tdo_en	O	-	JTAG	JTAG5 的 TD O 输出使能信号: 用于支持多个并行的 JTAG 控制器，当 tap 控制器处在 shift-dr 或 shift-ir 状态，并且 had_pad_jtg _tap_on 有效时，这个信号有效。
调试支持信号:				

下页继续

表 3.2 – 续上页

信号名	I/O	初始值	时钟域	功能描述
pad_core(x)_dbgrq_b	I	-	SYS	外部调试请求信号: 该信号是外部调试请求信号，低电平有效，同步于 syscl k，使能处理器核进入调试模式。 不用该信号时，需要接 1。
pad_core(x)_dbg_mask	I	-	SYS	核心 X 的调试屏蔽信号: 该信号可以屏蔽对核心 X 的调试请求，需要同步于 SYS 时钟域后接入。不用该信号时，需要接 0。
core(x)_pad_jdb_pm[1:0]	O	2’b00	CPU	各个处理器核工作模式指示信号: 这些输出信号表明处理器核的工作模式，异步于 pad _had_jtg_tclk。 00: normal 模式; 01: 低功耗模式; 10：调试模式; 11：保留。
pad_cpu(x)_hartid	I	-	SYS	核心 id 配置信号: 配置各个核心的 id。
低功耗系统集成:				
core(x)_pad_lpmd_b[1:0]	O	2’b11	SYS	各个处理器核低功耗模式状态信号: 当处理器核执行 wfi 指令时，core(x)_pad_lpmd_b[1: 0] 被相应的改变: 00: 低功耗模式; 11：Normal 模式。
cpu_pad_no_op	O	1’b0	SYS	L2 Cach e 空闲指示信号: 当各个处理器核都进入低功耗模式且 L2 Cache 无未完成的传输时，该信号有效。
pad_cpu_l2cache_flush_req	I	1’b0	SYS	L2Cache 清除请求信号，高电平有效。具体请看低功耗相关章节描述。
cpu_pad_l2cache_flush_done	O	1’b0	SYS	L2Cache 清除完成信号，高电平有效。具体请看低功耗相关章节描述。
pad_cpu_sleep_in	I	-	-	顶层上下电的请求信号。该信号在 CPU 内部没有任何逻辑。为给 SoC 的预留信号。
cpu_pad_sleep_out	O	-	-	顶层上下电的响应信号。该信号在 CPU 内部没有任何逻辑。为给 SoC 的预留信号。

下页继续

表 3.2 – 续上页

信号名	I/O	初始值	时钟域	功能描述
DFT 系统集成				
pad_yy_scan_enable	I	-	-	扫描使能: 扫描链的使能控制信号，高电平有效。 不用该信号时，需要接 0。
pad_yy_scan_mode	I	-	-	进入扫描模式: 使处理器进入扫描模式,此时的 cpu 时钟和系统时钟均为测试时钟, 只有处理器进入扫描模式,并且 pad_yy_sc an_enable 有效时，才可以通过扫描链进行测试。 不用该信号时，需要接 0。
pad_yy_icg_scan_en	I	-	-	SCAN 模式下 C910 内部 ICG 的扫描使能信号。在功能模式下该信号需要接 0。
pad_yy_scan_rst_b	I	-	-	SCAN 模式下的全局 Scan Reset，控制除分频逻辑外的全部寄存器。
pad_yy_dft_clk_rst_b	I	-	-	SCAN 模式下的 PLIC 分频逻辑的 Reset，需要接到芯片的 Power-Up Reset。SCAN 过程中不能复位。
pad_yy_mbist_mode	I	-	-	C910 内部 memory bist 模式的选择信号。在 bist 模式下 C910 的输入复位信号被绑定为 0。
pad_l2c_data_mbist_clk_ratio[2:0]	I	-	-	控制 MBIST 模式下 L2 Data Array 时钟分频比 000: 1 分频; 001: 2 分频; 010: 3 分频; 011: 4 分频; 100: 5 分频; 101: 6 分频; 110: 7 分频; 111: 8 分频;
pad_l2c_tag_mbist_clk_ratio[2:0]	I	-	-	控制 MBIST 模式下 L2 Tag/Dirty Array 时钟分频比 000: 1 分频; 001: 2 分频; 010: 3 分频; 011: 4 分频; 100: 5 分频; 101: 6 分频; 110: 7 分频; 111: 8 分频;
CPU 运行观测信号集成:				
core(x)_pad_retire0	O	1 ‘b0	CPU	各个处理器核指令 0 退休指示信号: 0：当前周期没有指令退休; 1：当前周期有指令退休。
core(x)_pad_retire0_pc[39:0]	O	-	CPU	各个处理器核退休指令 0 的 PC: 表明当前正在退休的指令的 PC。
core(x)_pad_retire1	O	1 ‘b0	CPU	各个处理器核指令 1 退休指示信号: 0：当前周期没有指令退休; 1：当前周期有指令退休。

下页继续

表 3.2 – 续上页

信号名	I/O	初始值	时钟域	功能描述
core(x)_pad_retire1_pc[39:0]	O	-	CPU	各个处理器核退休指令 1 的 PC: 表明当前正在退休的指令的 PC。
core(x)_pad_retire2	O	1 'b0	CPU	各个处理器核指令 2 退休指示信号: 0: 当前周期没有指令退休; 1: 当前周期有指令退休。
core(x)_pad_retire2_pc[39:0]	O	-	CPU	各个处理器核退休指令 2 的 PC: 表明当前正在退休的指令的 PC。
core(x)_pad_mstatus[63:0]	O	64' b-	CPU	各个处理器核的 MSTATUS 寄存器: 表示处理器当前状态的值, 具体参考 C910 用户手册。

第四章　时钟和复位

4.1　时钟概览

C910 的时钟树概览如 图 4.1 所示。

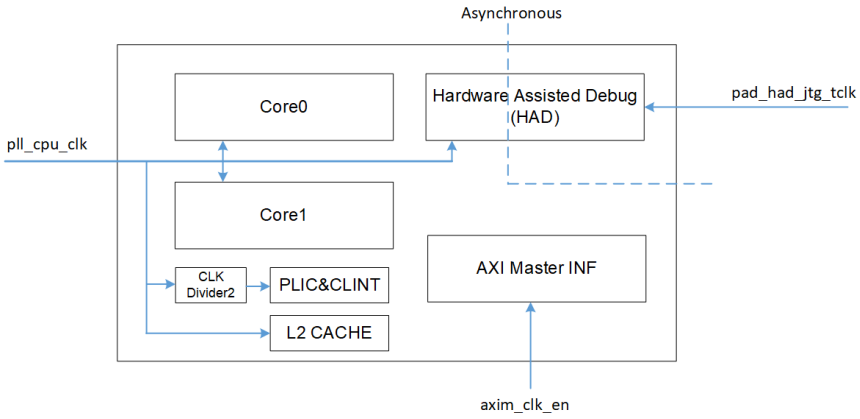


图 4.1: C910 的时钟概览

4.2　时钟信号

C910 CPU 有两个外部输入时钟，分别为：

- **pll_cpu_clk：**

  C910 的全局时钟，用于各个处理器核、L2 Cache 系统逻辑和中断控制器的工作时钟，其中中断控制器固定工作在 pll_cpu_clk 的二分频时钟。

- **pad_had_jtg_tclk：**

  JTAG 接口的工作时钟，用于 JTAG 状态机的运行，是一个异步于 pll_cpu_clk 的时钟。

  此外，处理器还有以下外部输入的同步时钟使能信号：

- **axim_clk_en：**

AXI 主设备接口所接总线与处理器的同步时钟使能信号，总线和处理器工作在同步时钟下，系统工作时钟与处理器工作时钟频率相等或者为处理器工作时钟频率的整数分频。系统总线和处理器进行交互时，根据 axim_clk_en 完成信号的同步。C910 支持系统时钟频率是处理器时钟频率的 1 到 8 分频。

4.3　时钟分频

C910 实现了时钟分频信号：axim_clk_en。当 CPU 与外围模块交互时，需要根据对应的时钟分频信号对输入输出信号进行采样，从而维护信号的同步。图 4.2 给出了 CPU 时钟和系统时钟在不同分频比下的的关系图。其中 clk_en（代指 axim_clk_en）高电平需要比 Sysclk 上升沿提早一个 cpu_clk 到达，clk_en 信号在 C910 内部使用 pll_cpu_clk 采样一拍后使用。

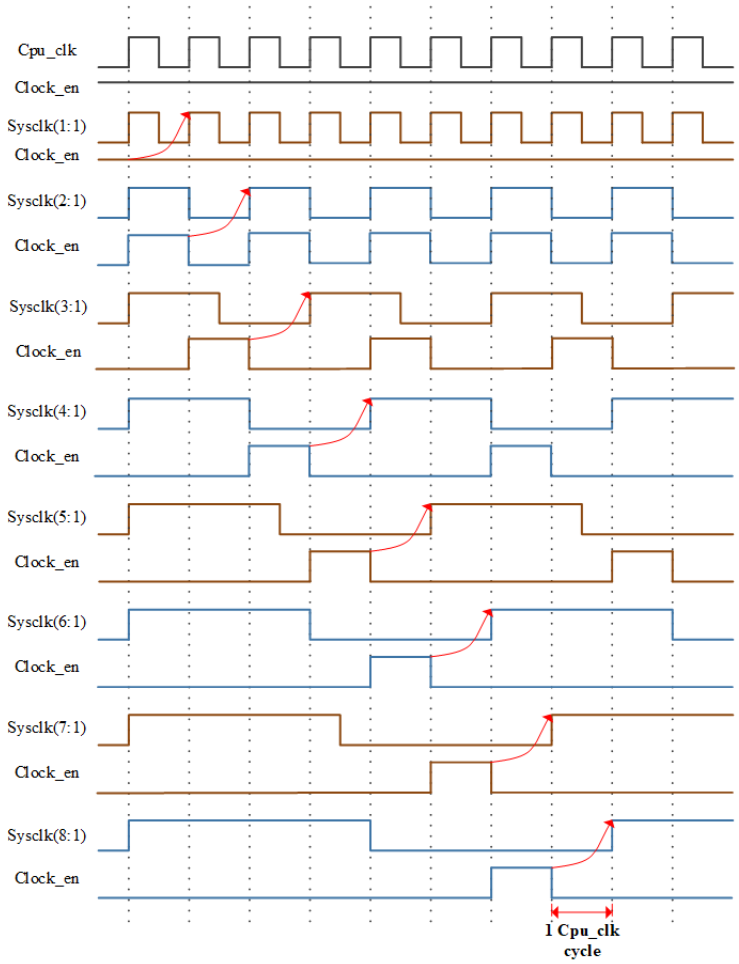


图 4.2: cpuclk, clk_en, sysclk 关系图

4.4　多时钟域信号同步

4.4.1　CPU 内核时钟域与系统总线时钟域

在 C910 的设计中，CPU 时钟域与系统总线（AXI 主设备接口）时钟域在总线接口部分逻辑中有信号交互。这三个时钟要求在物理设计中，需要保证为同步时钟。由于这三个时钟信号只有频率整数倍关系，所以无需用专用的逻辑进行同步。在 C910 的设计中，CPU 外部产生特定的控制信号（axim_clk_en），且 clk_en 信号在 CPU 内部作打拍处理（图 4.3 展示的是 clk_en 信号打拍后的时序图），保证 CPU 内部到总线上的所有信号都按照系统时钟的上升沿对齐。因此从 CPU 总线到系统的信号都具有 sysclk 的全周期时序延时。图 4.3 给出了 CPU 与系统时钟比例在 4：1 的情况下的 CPU 内部信号变化与总线接口上的信号变化。

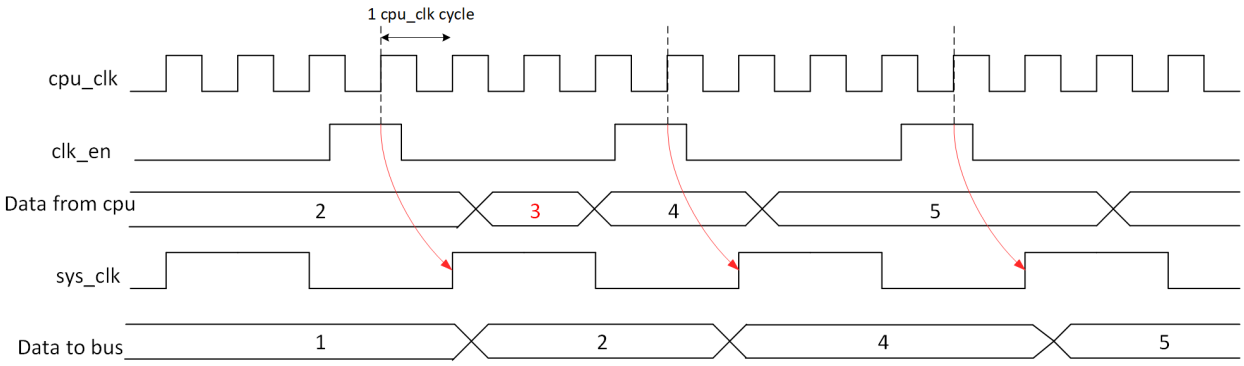


图 4.3: CPU 与系统时钟 4：1 情况下的接口信号

4.4.2　中断控制器时钟与内核时钟域

C910MP 内的 PLIC/CLINT 固定工作在 pll_cpu_clk 的二分频时钟下，分频逻辑由 C910MP 内部产生。因为在后端时序约束时需要将该部分逻辑和核内其他逻辑设置 Multi Cycle，具体请参考 C910 后端实现参考手册以及交付的 SDC 约束脚本。

4.4.3　L2 CACHE 与内核时钟域

当用户配置较大尺寸的 L2 Cache 或者选择功耗优先的 L2 Cache 时，L2 RAM 访问延时可能大于一个 CPU 时钟，此时需要软件在开启 L2 Cache 时通过设置 CCR2 寄存器来设置 RAM 的访问延迟周期。对应的在系统集成时，通过 set_multicycle_path 命令额外设置 RAM 数据出口信号的采样。具体示例如下：

```
create_generated_clock -name MEM0_CLK -source [get_ports pll_core_cpuclk] -edges {1 2 5}
[get_pins x_ct_l2c_top/x_ct_l2c_sub_bank_0/x_l2c_data/x_l2c_data_dout_gated_clk/x_gated_clk_cell/Q]

create_generated_clock -name MEM1_CLK -source [get_ports pll_core_cpuclk] -edges {1 2 5}
[get_pins x_ct_l2c_top/x_ct_l2c_sub_bank_1/x_l2c_data/x_l2c_data_dout_gated_clk/x_gated_clk_cell/Q]

set_multicycle_path -from CPUCLK -to MEM0_CLK -setup 2 -start

set_multicycle_path -from CPUCLK -to MEM0_CLK -hold 1 -start
```

set_multicycle_path -from CPUCLK -to MEM1_CLK -setup 2 -start

set_multicycle_path -from CPUCLK -to MEM1_CLK -hold 1 -start

注解： 1. 此示例是按照 L2 RAM 访问延时需要 2 个 CPU 时钟周期的假设进行设置；

2. 此示例中的信号名称和信号路径以实际的 SoC 集成方式替换；

3. 由于 L2 Cache 分 2 路，因此有 2 块独立的 RAM 需要设置。

4.4.4 CPU 内核时钟域与 JTAG 时钟域

在 C910 的设计中，CPU 内核时钟 cpuclk 与 JTAG 接口的 jtagclk 之间有信号交互。两个信号属于异步信号，用户无需关系他们之间的相位。CPU 内部已经通过同步逻辑单元将两个之间的信号进行了有效的三级同步。其逻辑框图如 图 4.4 所示。

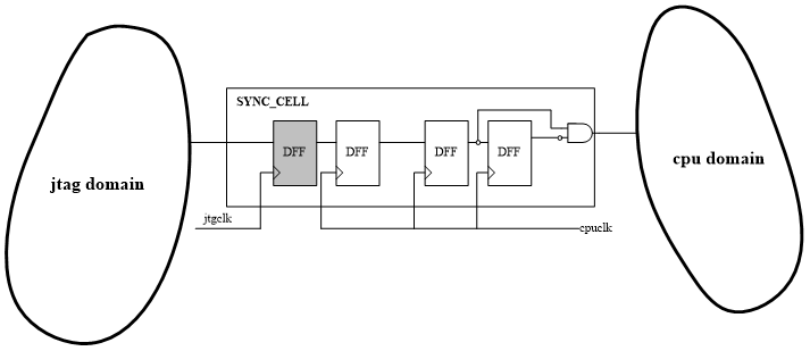


图 4.4: CPU 与 JTAG 时钟域之间的同步逻辑

4.5 复位信号与复位模式

C910 输入的复位信号如下：

- **pad_cpu_rst_b：**

  用于 C910 除核心之外所有模块的初始化，包含：L2 cache、中断控制器、JTAG 接口单元、master 接口；

- **pad_core(x)_rst_b：**

  为每个 C910 核心单独设置一根复位信号，用于各个核心独立维护初始化；

- **pad_had_jtg_trst_b：**

  用于 JTAG 接口的初始化；

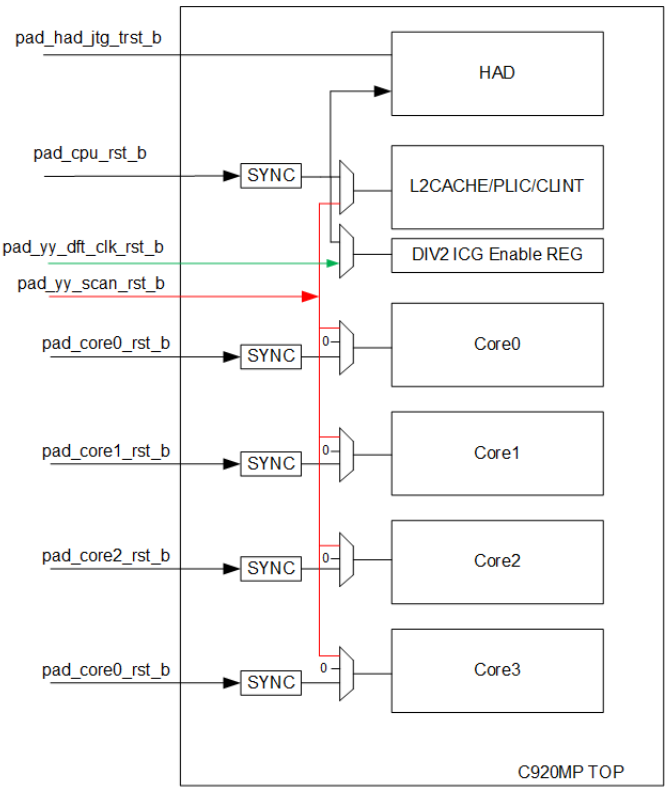


图 4.5: C910 复位信号概览

为了有效避免复位过程中亚稳态的出现，C910 采用异步复位、同步释放的方式进行系统的复位，因此需要各个复位信号同步到相对应的时钟域中。如上图所示，C910 的复位信号采用三级复位系统。例如各个核的复位信号在 CPU 时钟 pll_cpu_clk 的驱动下同步异步复位信号，可在复位信号释放时消除亚稳态。在 scan 模式下（pad_yy_scan_mode=1），用于同步的寄存器都被旁路，复位信号选择 pad_yy_scan_rst_b。

对于 cluster 顶层的调试单元（HAD COMMON TOP），部分调试寄存器的复位源为 pad_cpu_rst_b，JTAG 接口逻辑的复位源为 pad_had_jtg_trst_b。

注解： pad_had_jtag_trst_b 信号由外部仿真器完成 TCLK 时钟域的同步。

4.5.1 多核启动过程

C910 采用区分主次核的方式实现多核启动。SoC 可以选定 CORE0 为主核，SoC 释放 pad_core0_rst_b 控制主核退出 reset，等到主核完成系统初始化后 SoC 再相继释放其他核的复位信号，从而控制其他核退出 reset。

4.5.2 复位顺序

为了保证主核（假设是核 0）释放时，PLIC/L2CACHE/CIU 等单元处于确定状态，因此第一个释放核的复位信号 pad_core(x)_rst_b 在从 0 变成 1 时，pad_cpu_rst_b 要是 0 或者已经从 0 变为 1。为了保险起见，也可以将 pad_core(x)_rst_b 和 pad_cpu_rst_b 一起拉低，之后再提供时钟 pll_cpu_clk。

4.5.3 复位异常向量基地址可配置

C910 为每个核心提供了一组信号线 pad_core(x)_rvba[39:0] 供系统配置 CPU 的复位异常向量基地址。建议系统在集成时将该信号绑成固定值。

第五章　总线系统集成

5.1　C910 AXI 总线接口关键特性概述

AMBA 4.0 AXI 是一种高性能高频率的系统总线标准。C910 包含一个主设备接口，其总线协议与 AXI4.0 标准协议兼容。C910 的 AXI 总线接口以加速 CPU 访问片外存储器为目标而设计优化。

C910 主设备接口具有以下主要特征：

- 只支持以下 2 种传输方式：INCR（LEN 为 0 或者 3）和 WRAP4；
- 支持读的乱序完成；
- 支持关键字访问优先；
- 数据宽度为 128 位；
- 支持 CPU 到总线（cpu-to-bus）的整数倍分频（1:1, 2:1, 3:1, 4:1, 5:1, 6:1, 7:1, 8:1）；
- 支持各种总线响应：OKAY、EXOKAY、SLVERR 和 DECERR；
- 总线控制和数据信号寄存器输出时序；
- 支持独占（exclusive）原子操作；
- 不支持写操作的交叉（interleaving）；
- 不存在 4KB 边界问题。

5.2　AXI 主设备接口特性描述

C910 的 AXI 总线接口仅服务于 C910 体系结构，因此其 AXI 总线接口在兼容 AXI 4.0 协议标准的基础上仅实现了部分传输特性，表现在传输类型、宽度和长度上仅设计有若干特定属性。

5.2.1　读传输

5.2.1.1　读地址通道

C910 AXI 主设备接口读地址通道支持以下机制：

- 支持 INCR（长度为 1），WRAP4 两种传输类型，传输宽度支持 8、16、32、64、128；
- 系统级的 Cache 和 Buffer 控制；

读操作支持的 Outstanding 能力如下所示：

表 5.1: AXI 主设备接口读 Outstanding 能力

参数	数值	说明
Read Issuing Capability	8n+28 n= 核心数量	每个核心最多发出 8 个 Non-cacheable 和 Device 读请求。全局最多 28 个 Cacheable 读请求。

ARID 的编码如下所示：

表 5.2: AXI 主设备接口 ARID 编码

ARID[7:0]	适用场景	每个 ID 的 Outstanding
{2’ b10,6’ b??????}	Cacheable 读请求	每个 ID 无 outstanding，所有 cacheable 读请求 outstanding 总共 28 个。
{1’ b0,2’ b(coreid),5’ h18}	Non-cacheable weak-ordered 读请求	所有 Non-cacheable 读请求 outstanding 共 31 个。
{1’ b0,2’ b(coreid),5’ h1d}	Non-cacheable strong-ordered 读请求	

5.2.1.2　读数据通道

读数据通道承载从从设备返回的数据和响应两方面信息。数据通道特性包括：

- 数据总线宽度 128；
- 支持所有类型的读响应类型，OKAY、EXOKAY、SLVERR 和 DECERR；

5.2.2　写传输

C910 AXI 写传输请求的类型支持长度为 1 和 4 的 INCR，其中 INCR1 传输的宽度为字节/半字/字/双字/四字。不同宽度的数据传输时，对应的字节使能信号会有效。

5.2.2.1　写地址通道

写传输的地址通道包括以下特性：

- 只支持 INCR 一种传输类型；
- 传输宽度支持 8，16，32，64 和 128；
- 系统级的 Cache 和 Buffer 控制；

写传输的 Outstanding 能力如下所示：

表 5.3: AXI 主设备接口写 Outstanding 能力

参数	数值	说明
Write Issuing Capability	8n+32 n= 核心数量	每个核心最多发出 8 个 Non-cacheable 和 Device 写请求。全局最多 32 个 Cacheable 写请求。

AWID 的编码如下所示：

表 5.4: AXI 主设备接口 AWID 编码

AWID[7:0]	适用场景	每个 ID 的 Outstanding
{3’ b111,5’ b?????}	Cacheable 写请求	每个 ID 无 outstanding，所有 cacheable 写请求 outstanding 总共 32 个。
{4’ b0000,4’ b????}	Non-cacheable weak-ordered 写请求	每个 ID 无 outstanding，所有 Non-cacheable weak-ordered 写请求 outstanding 总共 16 个。
{1’ b0,2’ b(coreid),5’ h1d}	Non-cacheable strong-ordered 写请求	Non-cacheable strong-ordered 写请求 outstanding 为 31 个。

5.2.2.2　写数据通道

写数据通道的特性包括：

- 数据宽度 128 比特；
- 字节使能信号（byte lane strobe）指示字节有效性；

5.2.3　大小端

在 C910 处理器中，寄存器内部的值并没有大小端之分，只有有符号和无符号的区别。其格式均为从右至左表示逻辑低位到高位的排布，如 图 5.1 所示。

C910 的内存存放格式为高地址字节存放至物理内存的高位，即小端，如 图 5.2 所示。

5.2.4　AXI 主设备接口信号功能描述

下表所示为 AXI 主设备接口的信号描述。

表 5.5: 主结构信号列表

信号名	I/O	初始值	时钟域	功能描述
读地址通道信号				

下页继续

表 5.5 – 续上页

信号名	I/O	初始值	时钟域	功能描述
biu_pad_araddr[39:0]	O	40’b0	SYS	读地址总线： 40 位地址总线。
biu_pad_arburst[1:0]	O	2’b0	SYS	突发传输指示信号： 指示传输是一次突发传输的一部分： 00：SINGLE； 01：INCR； 10：WRAP4。
biu_pad_arcache[3:0]	O	4’b0	SYS	读请求对应的 cache 属性： [3]:Other Accocate； [2]:Allocate； [1]:Modifiable； [0]:Bufferable。
biu_pad_arid[7:0]	O	8’b0	SYS	读地址 ID。
biu_pad_arlen[7:0]	O	8’b0	SYS	突发传输长度： 00000000：1 拍; 00000011：4 拍。
biu_pad_arlock	O	1’b0	SYS	读请求对应的访问方式： 0：normal access； 1：exclusive access；
biu_pad_arprot[2:0]	O	3’b0	SYS	读请求的保护类型： 0 | 1 [2]: Data | Instruction； [1]: Secure | Non-secure； [0]: User | Privileged；
biu_pad_arsize[2:0]	O	3’b0	SYS	读请求每拍数据位宽： 000: 8bits; 001:16bits; 010:32bits; 011:64bits; 100: 128bits。
biu_pad_arvalid	O	1’b0	SYS	读地址有效信号。
pad_biu_arready	I	-	SYS	读地址通道 ready 信号。
读数据通道信号				
pad_biu_rdata[127:0]	I	-	SYS	读数据总线： 128 位数据总线。
pad_biu_rid[7:0]	I	-	SYS	读数据 ID。

下页继续

表 5.5 – 续上页

信号名	I/O	初始值	时钟域	功能描述
pad_biu_rresp[3:0]	I	-	SYS	读响应信号： [1:0]： 00：OKAY 01：EXOKAY； 10：SLVERR； 11：DECERR。
pad_biu_rlast	I	-	SYS	读数据最后一拍指示信号。
pad_biu_rvalid	I	-	SYS	读数据有效信号。
biu_pad_rready	O	1’b1	SYS	读数据通道 ready 信号。
写地址通道信号				
biu_pad_awaddr[39:0]	O	40’b0	SYS	写地址总线： 40 位地址总线。
biu_pad_awburst[1:0]	O	2’b0	SYS	突发传输指示信号： 指示传输是一次突发传输的一部分： 01：INCR； 10：WRAP4。
biu_pad_awcache[3:0]	O	4’b0	SYS	写请求对应的 cache 属性： [3]:Other Accocate； [2]:Allocate； [1]:Modifiable； [0]:Bufferable。
biu_pad_awid[7:0]	O	8’b0	SYS	写地址 ID。
biu_pad_awlen[7:0]	O	8’b0	SYS	突发传输长度： 00000000：1 拍; 00000011：4 拍。
biu_pad_awlock	O	1’b0	SYS	写请求对应的访问方式： 0：normal access； 1：exclusive access。
biu_pad_awprot[2:0]	O	3’b0	SYS	写请求的保护类型： 0 | 1 [2]: Data | Instruction； [1]: Secure | Non-secure； [0]: User | Privileged。

下页继续

表 5.5 – 续上页

信号名	I/O	初始值	时钟域	功能描述
biu_pad_awsize[2:0]	O	3’b0	SYS	写请求每拍数据位宽： 000: 8bits; 001:16bits; 010:32bits; 011:64bits; 100: 128bits。
biu_pad_awvalid	O	1’b0	SYS	写地址有效信号。
pad_biu_awready	I	-	SYS	写地址通道 ready 信号。
biu_pad_wdata[127:0]	O	128’b0	SYS	写数据总线： 128 位写数据总线。
biu_pad_wstrb[15:0]	O	16’b0	SYS	写数据字节有效信号。
biu_pad_wlast	O	1’b0	SYS	写数据最后一拍指示信号。
biu_pad_wvalid	O	1’b0	SYS	写数据有效信号。
pad_biu_wready	I	-	SYS	写数据通道 ready 信号。
写响应通道信号				
pad_biu_bid[7:0]	I	-	SYS	写响应 ID。
pad_biu_bresp[1:0]	I	-	SYS	写响应信号： 00：OKAY； 01：EXOKAY； 10：SLVERR； 11：DECERR。
pad_biu_bvalid	I	-	SYS	写响应有效信号。
biu_pad_bready	O	1’b1	SYS	写响应通道 ready 信号。

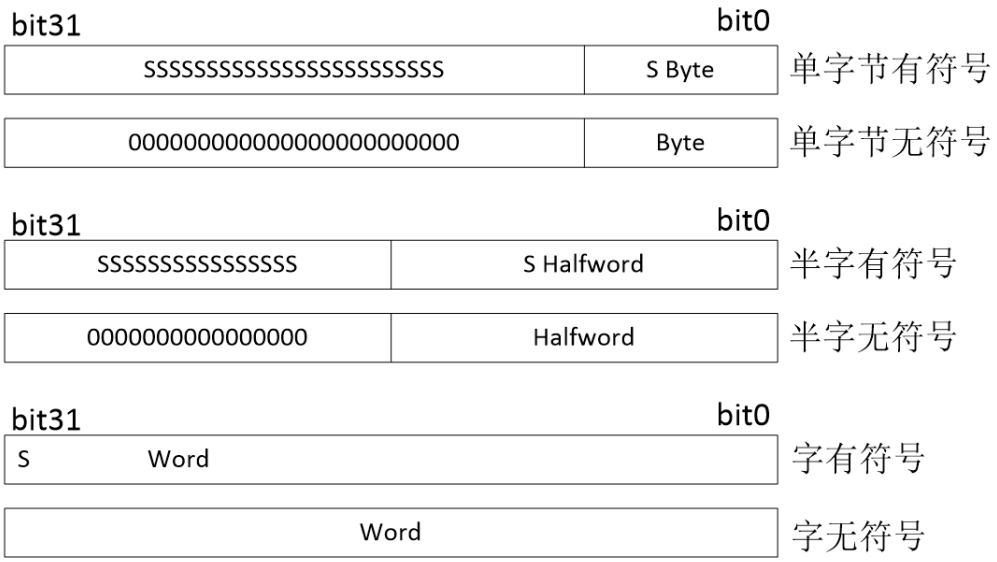


图 5.1: 数据在寄存器中的组织结构

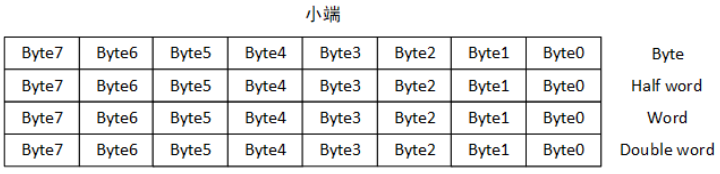


图 5.2: 数据在内存中的组织形式

第六章　中断系统集成

6.1　中断处理过程简述

中断处理是指处理器在接受到外部中断请求后从正常的程序处理转而响应中断处理，执行特定的中断处理程序。被中断的指令将正常退休，并在退休时响应中断请求。CPU 会保存当前的指令运行状态，将下一条指令作为中断返回的指令入口，并在退出中断服务程序时恢复之前的状态。C910 实现了 RISC-V 标准中断控制器 PLIC，支持将外设中断向多个 C910 核心进行中断的分发。

6.2　端口列表

内嵌中断控制器的接口信号主要用于 PLIC 接收并采样中断源的中断请求信号。

表 6.1: 中断控制器接口信号

信号名	I/O	初始值	时钟域	功能描述
pad_plic_int_vld[i-1:0]	I	-	SYS CLK	公有中断源的中断指示信号，表示公有中断源的中断是否有效。 1：表示中断有效； 0：表示中断无效。
pad_plic_int_cfg[i-1:0]	I	-	-	公有中断源的中断类型配置信号。 1：表示脉冲中断。 0：表示高电平中断。 该信号可以接常值，或者在非常值接入时需要由用户先将其同步到 pll_cpu_clk 时钟域。
pad_cpu_apb_base[39:0]	I	-	-	PLIC&CLINT 等内部中断控制寄存器的基地址指示信号，该信号需接常值且 bit[26:0] 需为 0，bit[39:27] 指定了上述控制寄存器地址空间的高 13 位。该寄存器空间一共占用 64MB，系统设计者在内存空间分配时需预留出来。
pad_cpu_sys_cnt[63:0]	I	-	PLIC	外部系统时钟，该信号需要由外部同步到片上 slave 时钟域

注解： 中断有效信号 pad_plic_int_vld[i-1:0] 接入 C910MP 后会被 pll_cpu_clk 时钟进行两级同步采样。pad_plic_int_vld[0] 的中断 ID 为 16，低 16 个中断 ID 被 C910MP 保留给内部使用（ID 为 0 的是无效中断）。

6.3　中断握手时序

接入 PLIC 的中断在 C910MP 内部进行采样和仲裁，在 PLIC 内部，仲裁逻辑工作在 pll_cpu_clk 的二分频时钟域。对于脉冲中断需要维持至少一个 pll_cpu_clk 的时钟宽度。电平中断要求中断服务程序中清除外设的中断源有效信号，否则当中断退出时会重新发起中断请求。对于脉冲中断，PLIC 采样中断有效信号的上升沿，然后根据中断的优先级向 CPU 发送中断请求，在 CPU 响应该脉冲中断请求前，若脉冲中断源向 PLIC 发起多次中断请求，PLIC 只会记录一次中断请求。在 CPU 响应该脉冲中断请求后，若脉冲中断源再次向中断控制器发起请求，PLIC 会再次采样对应中断，该中断请求在中断退出后才能够再次被 CPU 响应。

第七章　调试系统集成

7.1　端口列表

表 7.1: debug 系统端口列表

信号名	I/O	初始值	时钟域	功能描述
JTAG 支持信号				
pad_had_jtg_tclk	I	-	-	JTAG 测试时钟信号: JTAG 和 HAD 内部相关寄存器的时钟信号, 要求和 CPU 时钟的频率比在 1：8 以上。
pad_had_jtg_trst_b	I	-	TCLK	JTAG 复位信号
pad_had_jtg_tms	I	-	TCLK	JTAG 控制输入信号: JTAG 模式选择信号，控制 HAD 中 TAP 状态机的运作
pad_had_jtg_tdi	I	-	-	JTAG 数据输入信号: JTAG 串行输入端口,包括控制命令, 数据, 输入顺序由低位到高位。
had_pad_jtg_tdo	O	-	TCLK	JTAG 数据输出信号: JT AG 串行数据输出端口，输出顺序由低位到高位。
had_pad_jtg_tdo_en	O	-	TCLK	JTAG 的 TD O 输出使能信号: 用于支持多个并行的 JTAG 控制器，当 tap 控制器处在 shift-dr 或 shift-ir 状态, 这个信号有效。

表 7.2: debug 系统端口列表

信号名	I/O	初始值	时钟域	功能描述
调试支持信号				
pad_core(x)_dbgrq_b	I	-	SYS	外部调试请求信号: 该信号是外部调试请求信号，低电平有效，同步于 sysclk，使能处理器核进入调试模式。 不用该信号时，需要接 1。
core(x)_pad_jdb_pm[1:0]	O	2'b00	CPU	各个处理器核工作模式指示信号: 这些输出信号表明处理器核的工作模式，异步于 pad_had_jtg_tclk。 00: normal 模式; 01: 低功耗模式; 10：调试模式; 11：保留。
pad_core(x)_dbg_mask	I	-	SYS AXIM CLK	处理器核心 X 的禁止调试使能号: 1：禁止调试 coreX 0：可以调试 coreX
pad_core(x)_hartid[2:0]	I	-	-	该信号用于指示核的 ID 信号，核 0 接 0，以此类推。

7.2　JATG 接口

1. **pad_had_jtg_tclk**

   JTAG 时钟信号。该信号为外部输入信号，一般为调试器产生的时钟信号，要保证该时钟信号的频率低于 CPU 时钟信号的频率 1/8 才能保证调试模块与 CPU 核之间的正常工作。

2. **pad_had_jtg_trst_b**

   pad_had_jtg_trst_b 信号为 JTAG 复位信号，可以复位 TAP 状态机以及其他相关控制信号。

3. **JTAG5 相关信号**

   pad_had_jtg_tdi 信号为 HAD 端 JTAG 串行输入信号，HAD 端在 JTAG 时钟信号 tclk 的上升沿对其采样，而外部调试器在 JTAG 时钟的下降沿设置该信号;

   had_pad_jtg_tdo 信号为 HAD 端 JTAG 串行输出信号，HAD 端在 JTAG 时钟信号 tclk 的下降沿对其设置，而外部调试器在 JTAG 时钟的上升沿对其采样;

had_pad_jtg_tdo_en 信号表示 had_pad_jtg_tdo 信号有效，通常外部调试器监视该信号以确定 had_pad_jtg_tdo 信号是否有效，调试器也可以不观察该信号而通过调试器内部的 TAP 状态机的状态以决定对 had_pad_jtg_tdo 信号的采样。该信号主要用作在实现多个 TAP 状态机时，确定是哪个 JTAG 的输出；

pad_had_jtg_tms 信号为 JTAG 模式选择信号，该信号由调试器发出，用于控制 HAD 中 TAP 状态机的运作。

7.3　其他接口调试

T-HEAD CPU 还提供了事件触发的调试接口，可以使 CPU 通过非 JTAG 调试方式进入调试模式。如图 7.1 所示，SOC 设计时可以将某个事件作为 CPU 进入调试模式的触发条件，将该事件转化为 CPU 的调试请求信号 pad_corex_dbgrq_b。事件发生后该信号被置零，CPU 接收到请求进入调试模式。这样，调试人员就可在设定的 SOC 事件发生时使 CPU 进入调试模式。

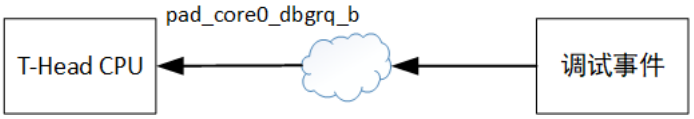


图 7.1: 事件触发调试

针对该功能可以有更灵活的设计，比如可以设计多个事件触发调试请求，每个事件设置相应的使能位，调试前进行相应的设置，就可以使该功能得到更丰富的扩展。

pad_core(x)_dbg_mask 信号用于禁止对 corex 进行调试的指示。当核心 x 处于下电流程中时 SoC 需要将该信号置 1，屏蔽对该核心的调试请求。

第八章　低功耗系统集成

8.1　端口列表

表 8.1: 低功耗端口列表

信号名	I/O	初始值	时钟域	功能描述
core(x)_pad_lpmd_b[1:0]	O	2’b11	SYS AXIM CLK	各个处理器核低功耗模式状态信号: 当处理器核执行 wfi 指令时, core(x)_pad_lpmd_b[1:0] 被相应的改变: 00: low power mode; 11：normal。
cpu_pad_no_op	O	1’b0	SYS AXIM CLK	L2 Cache 空闲指示信号: 当各个处理器核都进入低功耗模式且 L2 Cache 无未完成的传输时，该信号有效。
pad_cpu_l2cache_flush_req	I	-	SYS AXIM CLK	清除 L2 CACHE 的请求信号。该信号需要维持高电平状态直到 cpu_pad_l2cache_flush_done 信号为高电平。
cpu_pad_l2cache_flush_done	O	1’b0	SYS AXIM CLK	L2 CACHE 清除完成的指示信号。该信号在 pad_cpu_l2cache_flush_req 为高点平若干个时钟周期后变为高电平，之后系统集成者需要将 pad_cpu_l2cache_flush_r eq 信号拉低，下一个 SYS AXIM CLK 周期该信号变成低电平。

8.2　低功耗模式

C910MP 支持多种模式的低功耗策略，包括单核心的 clock gating，C910MP 顶层的 clock gating，单核心的 power gating 和整个 C910MP 的 power gating。通过由核心执行 WFI 低功耗指令并配合其他操作可完成上述的低功耗状态切换。

8.2.1　单核心的 Clock Gating

当 CPU 核心处于正常工作模式时，通过执行 WFI 指令该核心可以自动关闭核心内部绝大多数寄存器时钟，进入低功耗模式。当遇到调试请求，中断请求，复位，其他核发起的 snoop 请求时该核心自动退出低功耗模式。

8.2.2　顶层的 Clock Gating

当每个核心的 core(x)_pad_lpmd_b 为 2 ‘b00 且 cpu_pad_no_op 信号值为 1 时表示各个核心已处于低功耗模式且总线上无未完成的请求，SoC 可关闭 pll_cpu_clk 时钟，将整个 C910MP 切入 clock gating 的模式。

SoC 要将 C910MP 唤醒前需要先恢复稳定的 pll_cpu_clk 时钟。

8.2.3　单核心的 Power Gating

C910MP 支持单核心的下电操作，系统可通过关断核心电源完全关断核心的静态功耗。核心电源关断流程如下：

- 屏蔽核心的所有中断请求，包括外部中断、软中断和 timer 中断，关闭 MSTATUS/SSTATUS 寄存器中断使能位（mie、sie）和 MIE/SIE 寄存器中中断使能位
- 关闭数据预取
- 核心执行 INV&CLR D-Cache ALL，将 dirty line 写回 L2C
- 核心关闭 D-Cache（在清 cache 和关 cache 之间不能有 store 指令）
- 关闭核心 snoop enable 寄存器，屏蔽对该核心的 snoop
- 执行 fence iorw, iorw 指令
- 执行低功耗指令 WFI，核心进入低功耗模式

SoC 检测到核心低功耗输出信号（corex_pad_lpmd_b）有效时需继续完成如下单核心下电流程：

- 发送 pad_corex_dbg_mask 给对应 C910 核心，屏蔽调试请求
- 拉低 C910 待下电核心的复位信号 pad_corex_rst_b，复位待下电的 CPU 核心
- 激活核心输出信号 isolator 的 iso_en（该信号为 soc 通给 cpu，cpu 在 upf 里定义的 iso_en）
- 在确保 isolator 工作后，外部 PMU 关闭核心电源

核心在断电下只有通过复位才能重新启动核心。C910 仍在工作的核心检测到系统负载大，需要唤醒其他 CPU 核心时，SoC 执行以下操作：

- 仍在工作 C910 核心通过 mailbox 通知 SoC 需要唤醒的核心
- SoC 设置唤醒核心对应复位地址 rvbr（pad_corex_rvba）
- 拉低核心的复位信号（pad_corex_rst_b）
- 打开电源，保持复位信号不释放
- 释放核心输出信号 isolator 的 iso_en
- 释放核心复位信号，该核心启动
- 被唤醒核心执行初始化程序，开启 SMPEN 位，执行 MMU、DCACHE 使能等初始化操作。

8.2.4 顶层的 Power Gating

当 C910MP 各个核心电源关断后，顶层电源可选择关断，从而关闭整个 cluster 的静态功耗。顶层电源关断流程如下：

- Cluster 内除主核外其余三个核心电源均已关闭（这里以主核为例方便说明，最后一个下电的核心可以是任意核）

主核执行如下操作：

- 屏蔽核心的所有中断请求，包括外部中断、软中断和 timer 中断，关闭 MSTATUS/SSTATUS 寄存器中断使能位（mie、sie）和 MIE/SIE 寄存器中中断使能位。
- 关闭数据预取
- 执行 dcache inv&clr all 操作
- 关闭 Dcache
- 关闭核心的 SMPEN
- 执行 fence iorw，iorw 指令
- 执行低功耗指令 WFI，进入低功耗模式

SoC 系统在看到最后一个核心的 corex_pad_lpmd_b 信号有效时需执行如下操作：

- SoC 设置主核进入电源关断模式，屏蔽调试请求，激活主核输出信号 isolator 的 iso_en
- 外部 PMU 关闭主核电源
- 发送 pad_cpu_l2cache_flush_req 给 C910MP
- 等待 C910MP 返回 cpu_pad_l2cache_flush_done（SoC 在收到 cpu_pad_l2cache_flush_done 后拉低 pad_cpu_l2cache_flush_req，然后 cpu 会拉低 cpu_pad_l2cache_flush_done）
- 等待 C910MP 返回 cpu_pad_no_op，激活顶层输出信号 isolator 的 iso_en

- 关闭顶层电源

C910MP 顶层通过复位重新上电，上电流程如下：

- 拉低 cluster 内所有核心和顶层的复位信号
- 打开电源，保持复位信号不释放，pll_cpu_clk 时钟稳定
- 释放各核心和顶层的输出信号钳位 (iso_en)
- 释放各核心和顶层的复位信号
- 执行复位异常服务程序，恢复 CPU 状态。

备注：如果软件操作流程不符合上述执行顺序，结果不可预期。

另外，C910MP 顶层还保留一组信号 pad_cpu_sleep_in 和 cpu_pad_sleep_out。这两个信号在 CPU 内部没有任何逻辑，在掉电处理过程中 SoC 可以使用这两根信号作为顶层上下电的请求和响应信号。

8.3　退出低功耗模式握手

当某个核心处于 clock gating 的低功耗模式时，可通过以下方式唤醒：

1. 中断请求；
2. 调试请求；
3. 其他核发起的 snoop 请求。

当所有核心都处于低功耗模式时，只能由 ** 中断请求 ** 和 ** 调试请求 ** 唤醒 CPU 退出低功耗。

当 C910 将由中断唤醒时，可由其他核心设置 PLIC 产生中断唤醒；当处理器的中断使能位为低时，中断信号仅实现唤醒，并不会使该核心进入中断服务程序，唤醒之后 CORE 继续执行低功耗之后的指令，当中断使能位为高时，处理器将被唤醒，并进入中断服务程序。

当某个 CORE 接收到唤醒请求，顶层时钟门控会恢复该 CORE 的工作时钟 corex_clk。当所有 CORE 都在低功耗模式时，需要外部先恢复 CPU 全局时钟，然后再开启各个核的工作时钟。

第九章　DFT 系统集成

C910 的 RTL 集成了 DFT 相关信号，已具备 DFT 集成的先期条件，即通过相应的模式控制，在不同的测试模式下得到所需要的时钟和复位信号。C910 的 RTL 没有集成 MBIST 测试逻辑，系统设计人员需要通过 EDA 工具完成 MBIST 的集成工作。

C910 的 DFT 相关接口如 表 9.1 所述：

表 9.1: C910 DFT

信号名	I/O	功能描述
pad_yy_scan_mode	I	scan 模式选择信号。
pad_yy_scan_enable	I	C910MP 内部寄存器的 scan 使能信号。
pad_yy_scan_rst_b	I	scan 模式下的复位控制信号。
pad_yy_icg_scan_en	I	C910MP 内部 ICG 的 scan 使能信号。
pad_yy_dft_clk_rst_b	I	C910MP 内 PLIC/CLINT 的分频用 ICG 的使能寄存器的复位控制信号，该信号仅需接上电复位，scan 过程中该信号不可拉低。
pad_yy_mbist_mode	I	C910MP 内部 memory 的 bist 模式选择信号。
pad_l2c_data_mbist_clk_ratio	I	控制 bist 模式下 L2CACHE Data Array 时钟与 pll_cpu_clk 的分频比，该信号值的 0-7 对应时钟为 pll_cpu_clk 的 1-8 分频。
pad_l2c_tag_mbist_clk_ratio	I	控制 bist 模式下 L2CACHE TAF Array 时钟与 pll_cpu_clk 的分频比，分频逻辑与上述信号相同。

第十章　CPU 运行观测信号

CPU 运行观测信号是将处理器核内部的指令 dispatch 和指令 retire 信息输出到顶层供 SoC 来调试分析，SoC 设计可选择性的将这些信号锁存到外部寄存器中以方便调试，如果不用可悬空。

表 10.1: CPU 运行观测信号

信号名	I/O	初始值	时钟域	功能描述
core(x)_pad_retire0	O	1 ‘b0	CPU	处理器核心 X(X 值为 0-3，根据配置的处理器核心数而定) 的指令 0 退休指示信号: 0：当前周期没有指令退休; 1：当前周期有指令退休。
core(x)_pad_retire0_pc[39:0]	O	-	CPU	处理器核心 X 的退休指令 0 的 PC: 表明当前正在退休的指令的 PC。
core(x)_pad_retire1	O	1 ‘b0	CPU	处理器核心 X 的指令 1 退休指示信号: 0：当前周期没有指令退休; 1：当前周期有指令退休。
core(x)_pad_retire1_pc[39:0]	O	-	CPU	处理器核心 X 的退休指令 1 的 PC: 表明当前正在退休的指令的 PC。
core(x)_pad_retire2	O	1 ‘b0	CPU	处理器核心 X 的指令 2 退休指示信号: 0：当前周期没有指令退休; 1：当前周期有指令退休。
core(x)_pad_retire2_pc[39:0]	O	-	CPU	处理器核心 X 的退休指令 2 的 PC: 表明当前正在退休的指令的 PC。
core(x)_pad_mstatus[63:0]	O	-	CPU	处理器核心 X 的 MSTATUS 寄存器的当前值镜像。

第十一章　地址空间属性设置

系统设计人员需要对 C910 交付的代码包里的 sysmap.h 文件进行修改，以适配自己系统中关于地址空间属性的定义。该文件需要随 C910 一同放入客户的项目文件列表中。地址空间属性在设计阶段设定完成后，软件无法修改。

C910 支持两种内存类型，分别是内存（memory）和外设（device）（由 SO 位区分）。其中，memory 类型的特点是支持投机执行和乱序执行，根据是否可高缓（Cacheable，C）进一步分为可高缓内存（cacheable memory）和不可高缓内存（non-cacheable memory）。device 类型的特点为不可投机执行且必须按序执行，因此 device 一定带有不可高缓的属性。device 根据是否可缓存（Bufferable，B）分为可缓存外设（bufferable device）和不可缓存外设（non-bufferable device）（bufferable 表征 slave 能够快速返回写完成；反之，non-bufferable 表示 slave 只有在真正写完成后才返回写响应）。

为了支持多核之间数据共享，C910 增加了可共享的页面属性（Shareable，SH）。对于可共享的页面，表示该页面多核间共享，由硬件维护数据的一致性；对于不可共享的页面，表示被某个单核独占，不要求硬件维护数据的一致性。但如果不可共享的页面需要在多个核之间共享，则需要软件来维护数据的一致性。可高缓内存可以配置 SH 的属性，而不可高缓内存类型和外设类型的 SH 属性不可配置，固定为可共享。

另外，C910 支持配置安全的页面属性（Security，SEC）。

表 11.1 给出了各个内存类型对应的页面属性。页面属性的配置有两种方式：

- 在所有不进行虚拟地址和物理地址转换的情况下：机器模式权限或者 MMU 关闭，地址的页面属性由 sysmap.h 决定。sysmap.h 是 C910 扩展的，对用户开放，用户可以根据自身需求，定义不同地址段的页面属性，地址区域个数上限为 8。
- 在所有进行虚拟地址和物理地址转换的情况下：非机器模式权限且 MMU 打开时，地址的页面属性有两种配置方式：sysmap.h 和 C910 在 pte 中扩展的页面属性，具体取决于 C910 扩展寄存器 mxstatus 中的 maee 位是否打开。如果 maee 打开，地址的页面属性由对应 pte 中扩展的页面属性决定。如果 maee 关闭，地址的页面属性由 sysmap.h 决定。

表 11.1: 内存类型分类

内存类型	SO	C	B	SH	SEC
可高缓内存	0	1	1	可配	可配
不可高缓内存	0	0	1	1	可配
可缓存外设	1	0	1	1	可配
不可缓存外设	1	0	0	1	可配

sysmap.h：

支持对 8 个内存地址空间的属性设定，第 i（i 从 0 到 7）个地址空间地址上限（不包含）由宏 SYSMAP_BASE_ADDRi（i 从 0 到 7）定义，地址下限（包含）由 SYSMAP_BASE_ADDR(i-1) 定义，具体为：

SYSMAP_BASE_ADDR(i-1) <= 第 i 个地址空间地址 < SYSMAP_BASE_ADDRi

第 0 个地址空间下限是 0x0，内存地址不在 sysmap.h 文件设定的 8 个地址区间的地址属性默认为 weak order/cacheable/bufferable/shareable/security。每个地址空间上下边界是 4KB 对齐，因此宏 SYSMAP_BASE_ADDRi 定义的是地址的高 28 位。

落在第 i(i 从 0 到 7) 个地址空间内的地址的属性由宏 SYSMAP_FLAGi (i 从 0 到 7) 定义，属性的排布如 图 11.1 所示：

4	3	2	1	0
Strong order	cacheable	bufferable	Shareable	Security

图 11.1: sysmap.h 地址属性排布