



AN 796: Cyclone® V 和 Arria® V SoC 器件设计指南

针对 Intel® Quartus® Prime 设计套件的更新: **18.0**

本翻译版本仅供参考，如果本翻译版本与其英文版本存在差异，则以英文版本为准。某些翻译版本尚未更新对应到最新的英文版本，请参考[英文版本](#)以获取最新信息。



订阅

反馈

AN-796 | 2020.07.27

官网最新文档: [PDF](#) | [HTML](#)

内容

1. Cyclone® V SoC FPGA 和 Arria® V SoC FPGA 设计指南概述	4
1.1. SoC FPGA 设计人员核查表	5
1.2. SoC FPGA 设计中 HPS 设计指南概述	7
1.3. SoC FPGA 设计的电路板设计指南概述	8
1.4. SoC FPGA 设计的嵌入式软件设计指南概述	9
2. 背景: 对比 Cyclone V SoC FPGA 和 Arria V SoC FPGA HPS 子系统	10
2.1. HPS 和 FPGA 互连指南	10
2.1.1. HPS-FPGA 桥接	10
2.1.2. FPGA-to-HPS SDRAM 访问	12
2.1.3. 将软逻辑连接 HPS 组件	13
3. SoC FPGA 中 HPS 部分的设计指南	15
3.1. 由此开始 SoC-FPGA 设计	15
3.1.1. HPS-to-FPGA 接口设计的建议起点	15
3.1.2. 确定您的 SoC FPGA 拓扑	15
3.2. 连接器件 I/O 与 HPS 外设和存储器的设计考量	15
3.2.1. HPS 管脚分配设计考量	17
3.2.2. HPS I/O 设置: 约束和驱动强度	18
3.3. HPS 时钟和复位设计考量	19
3.3.1. HPS 时钟规划	19
3.3.2. 早期管脚规划与 I/O 约束分析	19
3.3.3. HPS JTAG, 时钟, 复位和 PoR 的管脚功能和连接	19
3.3.4. 内部时钟	20
3.4. HPS EMIF 设计考量	20
3.4.1. HPS 连接 SDRAM 的考量	20
3.4.2. HPS SDRAM I/O 位置	22
3.4.3. HPS EMIF 与 SoC FPGA 器件集成	22
3.4.4. HPS 存储器调试	22
3.4.5. HPS 地址镜像技术	22
3.5. DMA 考量	24
3.5.1. 选择 DMA 控制器	24
3.5.2. 通过 HPS 互连优化 DMA 主控带宽	24
3.5.3. FPGA 加速器的时序收敛	24
3.6. 管理 FPGA 加速器一致性	24
3.6.1. 高速缓存一致性	25
3.6.2. FPGA 逻辑和 HPS 之间的一致性: 加速器一致性端口 (ACP)	25
3.6.3. 数据大小影响 ACP 性能	25
3.6.4. 避免 ACP Dependency 锁定	25
3.6.5. FPGA 通过 AXI 或 Avalon-MM 对 ACP 的访问	25
3.6.6. ACP 和 L2 Cache ECC 访问的数据对齐	25
3.7. IP 调试工具	26
4. SoC FPGA 的电路板设计指南	27
4.1. 电路板启动考量	27

4.1.1. 保留的 BSEL 设置.....	27
4.2. 引导和配置设计考量.....	27
4.2.1. 引导设计考量.....	27
4.2.2. 配置.....	30
4.2.3. 参考资料.....	30
4.3. HPS 电源设计考量.....	31
4.3.1. 早期系统和电路板规划.....	31
4.3.2. SoC FPGA 器件 HPS 和 FPGA Power Supplies 的设计考量.....	32
4.3.3. 电路板设计的管脚连接考量.....	32
4.3.4. 功率分析和优化.....	33
4.4. HPS 的边界扫描.....	34
4.5. HPS 接口的设计指南.....	34
4.5.1. HPS EMAC PHY 接口.....	34
4.5.2. USB 接口设计指南.....	39
4.5.3. QSPI Flash 接口设计指南.....	40
4.5.4. SD/MMC 和 eMMC 卡接口设计指南.....	41
4.5.5. NAND Flash 接口设计指导.....	42
4.5.6. UART 接口设计指南.....	42
4.5.7. I ² C 接口设计指南.....	43
4.5.8. SPI 接口设计指南.....	43
5. SoC FPGA 的嵌入式软件设计指南.....	45
5.1. HPS 的嵌入式软件：设计指南.....	45
5.1.1. 组装软件开发平台的组件.....	45
5.1.2. 为应用程序选择操作系统.....	48
5.1.3. 组装用于 Linux 的 Software Development Platform.....	49
5.1.4. 组装 Bare-Metal 应用程序的 Software Development Platform.....	53
5.1.5. 组装用于合作伙伴 OS 或 RTOS 的软件开发平台.....	54
5.1.6. 选择引导加载程序软件.....	54
5.1.7. 选择用于开发，调试和跟踪的软件工具。.....	55
5.2. Flash 器件驱动设计考量.....	56
5.3. SD 卡低功耗模式设计考量.....	56
5.4. HPS ECC 设计考量.....	57
5.4.1. 常规 ECC 设计考量.....	57
5.4.2. 系统级 ECC 控制，状态和中断管理.....	57
5.4.3. L2 高速缓存数据存储器 ECC.....	57
5.4.4. Flash Memory 的 ECC.....	58
5.5. HPS SDRAM 考量.....	58
5.5.1. 使用 Preloader 调试 HPS SDRAM.....	58
5.5.2. 通过 FPGA-to-SDRAM 接口访问 HPS SDRAM.....	63
A. 支持和文档.....	64
A.1. 支持.....	64
A.2. 软件文档.....	65
B. 附加信息.....	66
B.1. Cyclone V 和 Arria V SoC 器件指南修订历史.....	66



1. Cyclone® V SoC FPGA 和 Arria® V SoC FPGA 设计指南概述

本文档旨在对使用 Cyclone V SoC 和 Arria V SoC FPGA 器件的设计提供一组设计指南，建议以及一系列考虑因素。本文档有助于规划 SoC FPGA 设计，Platform Designer (Standard) 子系统设计，电路板设计和软件应用程序设计并对早期设计阶段提供帮助。

注意:

本应用笔记不包括全部 Cyclone V/ Arria V Hard Processor System (HPS) 器件详情，功能特性或设计硬件或软件系统的信息。有关 Cyclone V 或 Arria V HPS 功能特性和各个外设的更多信息，请参阅相应的 Hard Processor System Technical Reference Manual (硬核处理器系统技术参考手册)。

*Arria V 和 Cyclone V 设计指南*中提供对设计中 FPGA 部分的设计指导。

相关链接

- [Arria V 硬核处理器系统技术参考手册](#)
- [Cyclone V 硬核处理器系统技术参考手册](#)
- [Intel MAX 10 FPGA 设计指南](#)

1.1. SoC FPGA 设计人员核查表

表 1. SoC FPGA 设计人员核查表

步骤标题	链接	核查情况 (X)
HPS 设计人员 SoC FPGA 核查表		
开始 SoC FPGA 设计	由此开始 SoC-FPGA 设计 (第 15 页)	
	确定您的 SoC FPGA 拓扑 (第 15 页)	
连接器件 I/O 与 HPS 外设和存储器的设计考量	HPS 管脚分配设计考量 (第 17 页)	
	HPS I/O 设置: 约束和驱动强度 (第 18 页)	
HPS 时钟和复位设计考量	HPS 时钟规划 (第 19 页)	
	早期管脚规划与 I/O 约束分析 (第 19 页)	
	HPS JTAG, 时钟, 复位和 PoR 的管脚功能和连接 (第 19 页)	
	内部时钟 (第 20 页)	
HPS EMIF 设计考量	HPS 连接 SDRAM 的考量 (第 20 页)	
	HPS SDRAM I/O 位置 (第 22 页)	
	HPS EMIF 与 SoC FPGA 器件集成 (第 22 页)	
	HPS 存储器调试 (第 22 页)	
DMA 考量	选择 DMA 控制器 (第 24 页)	
	通过 HPS 互连优化 DMA 主控带宽 (第 24 页)	
	FPGA 加速器的时序收敛 (第 24 页)	
管理 FPGA 加速器的一致性	高速缓存一致性 (第 25 页)	
	FPGA 逻辑和 HPS 之间的一致性: 加速器一致性端口 (ACP) (第 25 页)	
	数据大小影响 ACP 性能 (第 25 页)	
	FPGA 通过 AXI 或 Avalon-MM 对 ACP 的访问 (第 25 页)	
	ACP 和 L2 Cache ECC 访问的数据对齐 (第 25 页)	
IP 调试工具	IP 调试工具 (第 26 页)	
电路板设计人员的 SoC FPGA 核查表		
HPS 电源设计考量	早期系统和电路板规划 (第 31 页)	
	早期功耗估算 (第 31 页)	
	SoC FPGA 器件 HPS 和 FPGA Power Supplies 的设计考量 (第 32 页)	
	电路板设计的管脚连接考量 (第 32 页)	
	器件上电 (第 32 页)	
	功率分析和优化 (第 33 页)	
HPS 边界扫描	HPS 的边界扫描 (第 34 页)	
HPS 接口的设计指南	HPS EMAC PHY 接口 (第 34 页)	

继续...

步骤标题	链接	核查情况 (X)
	USB 接口设计指南 (第 39 页)	
	QSPI Flash 接口设计指南 (第 40 页)	
	SD/MMC 和 eMMC 卡接口设计指南 (第 41 页)	
	NAND Flash 接口设计指南 (第 42 页)	
	UART 接口设计指南 (第 42 页)	
	I2C 接口设计指南 (第 43 页)	
	SPI 接口设计指南 (第 43 页)	
嵌入式软件设计人员的 SoC FPGA 嵌入式的核查表		
组装 Software Development Platform 组件	组装软件开发平台的组件 (第 45 页)	
	Golden Hardware Reference Design (第 46 页)	
为应用程序选择操作系统(OS)	Linux 或 RTOS (第 48 页)	
	Bare Metal (裸机) (第 48 页)	
	使用 Symmetrical 与 Asymmetrical Multiprocessing (SMP vs. AMP) 模式 (第 48 页)	
组装 Linux Software Development Platform	面向 Linux 的黄金系统参考设计 (GSRD) (第 49 页)	
	GSRD for Linux 开发流程 (第 51 页)	
	GSRD for Linux 创建流程 (第 51 页)	
	Linux Device Tree 设计考量 (第 52 页)	
组装 Software Development Platform 用于 Bare-metal Application	组装 Bare-Metal 应用程序的 Software Development Platform (第 53 页)	
组装 Software Development Platform 用于 Partner OS/RTOS Application	组装用于合作伙伴 OS 或 RTOS 的软件开发平台 (第 54 页)	
选择引导加载程序软件	选择引导加载程序软件 (第 54 页)	
选择用于开发, 调试和跟踪的软件工具。	选择软件构建工具 (第 55 页)	
	选择软件调试工具 (第 56 页)	
	选择软件跟踪工具 (第 56 页)	
电路板启动考量	电路板启动考量 (第 27 页)	
引导和配置设计考量	引导设计考量 (第 27 页)	
	配置 (第 30 页)	
Flash 器件驱动器考量	Flash 器件驱动设计考量 (第 56 页)	
HPS ECC 设计考量	HPS ECC 设计考量 (第 57 页)	
HPS SDRAM 考量	HPS SDRAM 考量 (第 58 页)	

1.2. SoC FPGA 设计中 HPS 设计指南概述

表 2. HPS 设计指南概述

HPS 设计流程的各个阶段	指南	连接
硬件和软件分区	确定系统拓扑并将其用作 HPS 到 FPGA 接口设计的起点。	HPS 和 FPGA 互连指南 (第 10 页)
HPS 管脚复用和 I/O 配置设置	规划 HPS 系统的配置设置, 包括 I/O 复用选项, FPGA 和 SDRAM 接口, 时钟, 外设设置。	连接器件 I/O 与 HPS 外设和存储器的设计考量 (第 15 页)
HPS 时钟和复位考量	HPS 时钟和冷复位/热复位考量	HPS 时钟和复位设计考量 (第 19 页)
HPS EMIF 考量	HPS EMIF 控制器的使用和相关考量	HPS EMIF 设计考量 (第 20 页)
FPGA 加速器设计考量	管理 FPGA 加速器和 HPS 之间一致性的设计考量	DMA 考量 (第 24 页)
建议的 IP 开发工具	Signal Tap II, BFM, System Console	IP 调试工具 (第 26 页)

1.3. SoC FPGA 设计的电路板设计指南概述

表 3. 电路板设计：设计指南概述

电路板设计流程的阶段	指南	链接
HPS 电源设计考量	板上电源启动，早期功耗评估，HPS 和 FPGA 电源，功耗分析和功耗优化的设计考量。	HPS 电源设计考量 (第 31 页)
HPS 接口的电路板设计指南	包括 EMAC, USB, QSPI, SD/MMC, NAND, UART 和 I ² C	HPS 接口的设计指南 (第 34 页)

1.4. SoC FPGA 设计的嵌入式软件设计指南概述

表 4. 嵌入式软件：设计指南概述

嵌入式软件设计流程的各个阶段	指南	链接
操作系统 (OS) 考量	OS 考量旨在满足您应用程序的需求，包括实时性，软件复用，支持和易用性考量。	为应用程序选择操作系统 (第 48 页)
引导加载程序考量	引导加载程序考量旨在满足您应用程序的需求，包括 GPL 要求和功能。	选择引导加载程序软件 (第 54 页)
引导和配置设计考量	引导源，引导时钟，引导熔丝，配置流程	引导和配置设计考量 (第 27 页)
HPS ECC 考量	ECC 用于外部 SDRAM 接口，L2 cache 数据存储器，flash 存储器	HPS ECC 设计考量 (第 57 页)
HPS SDRAM 考量	使用 Preloader 调试 HPS SDRAM，访问 HPS SDRAM	HPS SDRAM 考量 (第 58 页)

2. 背景：对比 Cyclone V SoC FPGA 和 Arria V SoC FPGA HPS 子系统

Cyclone V SoC 和 Arria V SoC 中的 HPS 子系统在架构上相似，以下列出功能中存在的差异。

HPS 功能	Cyclone V SoC	Arria V SoC
Maximum MPU Frequency	最高达到 925 MHz	最高达到 1.05 GHz
Controller Area Network (CAN)	Yes	No
Total HPS Dedicated I/O with Loaner capability	最高达到 67 个	94 ⁽¹⁾
Automotive Grade Option	Yes	No
HPS SDRAM 支持的最大 DDR3 频率	400 MHz	533 MHz

[相关链接](#)

[Intel SoC 器件系列之间的差异](#)

2.1. HPS 和 FPGA 互连指南

HPS 和 FPGA 逻辑架构之间的存储器映射连接性是最大限度提高设计性能的关键工具。

*Arria V 和 Cyclone V 设计指南*中提供对设计中 FPGA 部分的设计指导。

[相关链接](#)

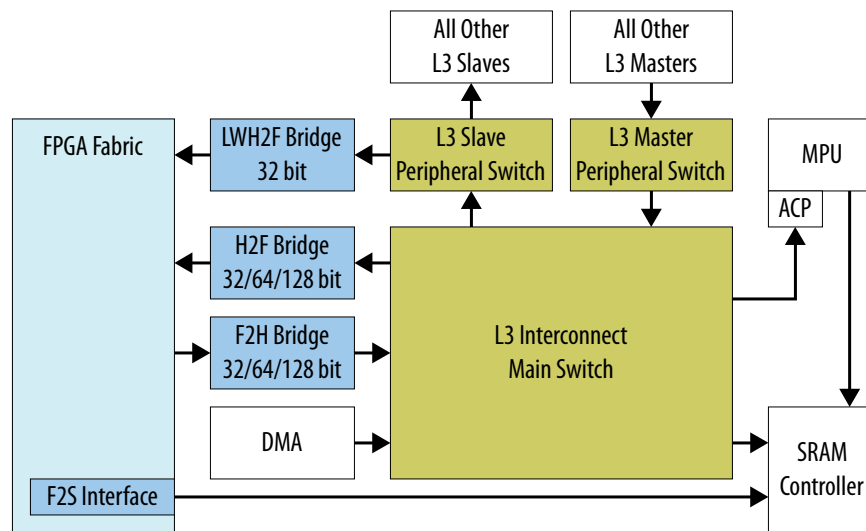
[Arria V 和 Cyclone V 设计指南](#)

2.1.1. HPS-FPGA 桥接

HPS 有三个桥接，他们使用存储器映射接口连接到基于 Arm* Advanced Microcontroller Bus Architecture (AMBA*) Advanced eXtensible Interface (AXI*) 的 FPGA。对这些桥接的使用决定了每个桥接的方向。

⁽¹⁾ 最多只能将 71 个 HPS I/O 作为 Loaner I/O 分配给 FPGA。有关 Cyclone V SoC 和 Arria V SoC HPS 子系统之间的详细比较，请参阅 [Intel SoC 器件系列之间的差异](#)。

图 1. HPS-FPGA 桥接



关键字:

H2F: HPS-to-FPGA

LWH2F: Lightweight HPS-to-FPGA

F2H: FPGA-to-HPS

F2S: FPGA-to-SDRAM

2.1.1.1. 轻量 HPS-to-FPGA 桥接

指南: 使用轻量 HPS-to-FPGA 桥接连接需要 HPS 控制的 IP。

轻量 HPS-to-FPGA 桥接允许 HPS 中的主接口访问 SoC 器件 FPGA 部分中的存储器映射控制从端口。通常, 仅 HPS 中的 MPU 访问该桥接以执行对 FPGA 中外设的控制和状态寄存器访问。

指南: 请勿将轻量级 HPS-to-FPGA 桥接用于 FPGA 存储器。但可以将 HPS-to-FPGA 桥接用于存储器。

当 MPU 访问外设内的控制和状态寄存器时, 其中事务为强有序 (non-posted)。将轻量 HPS-to-FPGA 桥接专用于寄存器访问时, 访问时间被最小化, 因为突发流量被路由到 HPS-to-FPGA 桥接。

轻量 HPS-to-FPGA 桥接通过固定 32-bit 宽连接到 FPGA 逻辑, 因为大多数 IP 核实现 32-bit 控制和状态寄存器。然而, Platform Designer (Standard) 也适应 FPGA 生成的网络互连中 32 位宽度以外的事务。

2.1.1.2. HPS-to-FPGA 桥接

指南: 使用 HPS-to-FPGA 桥接将由 FPGA 加载的存储器连接到 HPS。

HPS-to-FPGA 桥接允许 HPS 中的主接口 (如, 微处理器单元 (MPU), DMA 或集成了主接口的外设) 访问 SoC 器件中由 FPGA 部分加载的存储器。该桥接支持 32-bit、64-bit 和 128-bit 数据路径, 从而可将与桥接连接的 FPGA 逻辑中从数据宽度调整到最大。计划由主接口使用该桥接执行突发传输, 且不应用于访问 FPGA 逻辑中的外设寄存器。控制和状态寄存器访问应该发送到轻量级 HPS-to-FPGA 桥接。

指南: 如果使用连接 **HSP-to-FPGA** 桥接的存储器进行 **HPS** 引导, 请确保 **Platform Designer (Standard)** 中其从接口地址设置为 **0x0**。

设置 HPS BSEL 管脚从 FPGA (BSEL = 1) 引导, 处理器执行位于 HPS-to-FPGA 桥接中偏移 0x0 处由 FPGA 加载的编码。这是唯一可在引导时加载编码的桥接。

2.1.1.3. FPGA-to-HPS 桥接

指南: 使用 **FPGA-to-HPS** 桥接进行 **FPGA** 主接口到 **HPS** 的可高速缓存访问。

FPGA-to-HPS 桥接允许 FPGA 逻辑中实现的主接口访问 HPS 内的存储器和外设。该桥接支持 32, 64 和 128-bit 数据路径, 因此可将其调整到与 FPGA 中实现的主接口一样宽。

指南: 使用 **FPGA-to-HPS** 桥接访问高速缓存一致性存储器, 外设, 或进行从 **FPGA** 主接口到 **HPS** 中片上 **RAM** 的访问。

尽管该桥接有对 SDRAM 子系统的直接连接, 但该桥接的主旨是为外设和片上存储器提供访问权限, 以及提供与 MPU 加速器一致性端口 (ACP) 连接一致性。

在没有一致性的情况下访问 HPS, 应该将 FPGA 中的主接口连接到 FPGA-to-SDRAM 端口, 因为他们能够提供更多带宽和更低延迟的访问。

2.1.2. FPGA-to-HPS SDRAM 访问

除了 FPGA-to-HPS 桥接之外, FPGA 逻辑还可以使用 FPGA-to-SDRAM 接口访问 HPS SDRAM。

指南: 使用 **FPGA-to-SDRAM** 端口进行 **FPGA** 主接口对 **HPS SDRAM** 的非可高速缓存访问。

FPGA-to-SDRAM 端口允许 FPGA 逻辑中实现的主接口直接访问 HPS, 无需通过 L3 互连传输事务流程。

这些接口仅连接 HPS SDRAM 子系统, 因此如果 FPGA 需要对 HPS SDRAM 进行高吞吐, 低延迟访问, 则建议设计中使用这些接口。除非 FPGA 要求对 SDRAM 进行高速缓存一致性访问, 可以不参考该建议。

FPGA-to-SDRAM 接口不能访问 MPU ACP 从接口, 因此, 如果你需要 FPGA 逻辑中实现的主接口访问高速缓存一致的数据, 请确保该接口连接到 FPGA-to-HPS 桥接。

FPGA-to-SDRAM 接口具有三种端口类型, 用于创建 AXI 和 Avalon-MM 接口:

- 命令端口—发布读和写命令, 以及接收写确认响应
- 64-bit 读数据端口—接收存储器读取中返回的数据
- 64-bit 写数据端口—发送写数据

最多 6 个命令端口, 4 个 64-bit 读数据端口和 4 个 64-bit 写数据端口。下表显示可能的端口利用情况。

表 5. FPGA-to-HPS SDRAM 端口利用情况

总线协议	命令端口	读数据端口	写数据端口
32- or 64-bit AXI	2	1	1
128-bit AXI	2	2	2
256-bit AXI	2	4	4
32- or 64-bit Avalon-MM	1	1	1
128-bit Avalon-MM	1	2	2
256-bit Avalon-MM	1	4	4
32- or 64-bit Avalon-MM write-only	1	0	1
128-bit Avalon-MM write-only	1	0	2
256-bit Avalon-MM write-only	1	0	4
32- or 64-bit Avalon-MM read-only	1	1	0
128-bit Avalon-MM read-only	1	2	0
256-bit Avalon-MM read-only	1	4	0

有关 FPGA-to-HPS SDRAM 接口的更多信息, 请参阅 *Cyclone V or Arria V SoC Hard Processor System Technical Reference Manual* 中的“SDRAM 控制器子系统部分”章节。

注意: 要通过 FPGA-to-SDRAM 接口访问 HPS SDRAM, 请遵循通过 [FPGA-to-SDRAM 接口访问 HPS SDRAM](#) (第 63 页)中的指南, 如下。

相关链接

- [SDRAM Controller Subsystem - Cyclone V Hard Processor System Technical Reference Manual](#)
- [SDRAM Controller Subsystem - Arria V Hard Processor System Technical Reference Manual](#)

2.1.3. 将软逻辑连接 HPS 组件

设计人员可使用 Platform Designer (Standard)中的 Cyclone V/ Arria V HPS 组件将软逻辑组件连接到 HPS。

注意: 请参阅相应 *Hard Processor System Technical Reference Manual* 中的“HPS 组件介绍”和“例化 HPS 组件”章节了解接口和可用选项。将 FPGA 软 IP 组件连接到 HPS, Platform Designer (Standard)会提供组件编辑工具。请参阅 *Intel® Quartus® Prime Standard Edition 手册, 第一卷: 设计和综合*中的“[创建 Platform Designer \(Standard\)组件](#)”章节了解更多信息。

注意: 在 FPGA 内核中设计和配置高带宽 DMA 主控以及相关缓冲器时, 请参阅本文档的 [DMA 考量](#) (第 24 页)小节。上述部分涵盖的原则适用于所有高带宽 DMA 主控 (例如, Platform Designer (Standard) DMA Controller 组件, 定制外设中的集成 DMA 控制器) 和 FPGA 内核中通过 FPGA-to-SDRAM 和 FPGA-to-HPS 桥接端口访问 HPS 源 (例如, HPS SDRAM) 的相关缓冲, 而非紧密耦合 Arm CPU 硬件加速器。

相关链接

- [Introduction to the HPS Component - Cyclone V Hard Processor System Technical Reference Manual](#)
- [Instantiating the HPS Component - Cyclone V Hard Processor System Technical Reference Manual](#)
- [Introduction to the HPS Component - Arria V Hard Processor System Technical Reference Manual](#)
- [Instantiating the HPS Component - Arria V Hard Processor System Technical Reference Manual](#)

3. SoC FPGA 中 HPS 部分的设计指南

3.1. 由此开始 SoC-FPGA 设计

3.1.1. HPS-to-FPGA 接口设计的建议起点

根据您的拓扑结构，可选择两个硬件参考设计的其中之一作为您硬件设计的起点。

指南：使用 [Golden System Reference Design \(GSRD\)](#) 作为松散耦合系统的起点。

可将 Golden Hardware Reference Design (GHRD) 的最佳默认设计和时序用作您“入门”系统的基础。初步评价后，就可接着了解 Cyclone V HPS-to-FPGA Bridge Design Example 参考实例，以比较各种 FPGA-HPS 接口的性能。

请参阅“Golden Hardware Reference Design”了解更多信息。

指南：使用 [Cyclone V HPS-to-FPGA Bridge Design Example](#) 参考设计来确定 FPGA 逻辑和 HPS 之间访问的最佳突发长度和数据宽度。

Cyclone V FPGA-to-HPS 桥接设计包含 FPGA 逻辑中生成的模块化 SGDMA，从而允许对 FPGA 逻辑和 HPS 之间数据访问的突发长度进行编程。

相关链接

[Golden Hardware Reference Design](#) (第 46 页)

3.1.2. 确定您的 SoC FPGA 拓扑

要确定最适合于您应用程序的系统拓扑，必须首先确定如何划分应用程序硬件和软件分区。

指南：分析您的软件，确定实现硬件加速的功能。

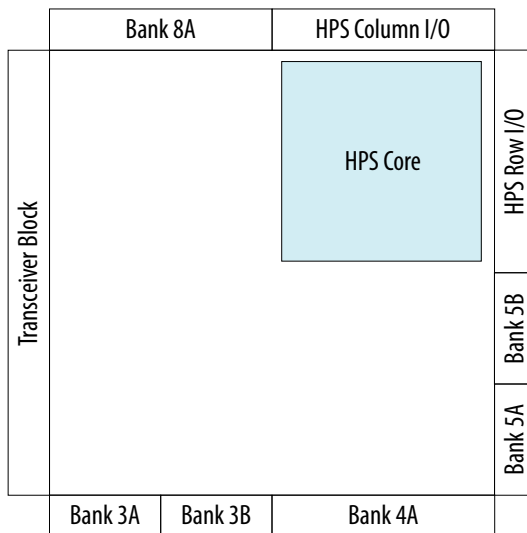
使用良好的性能分析工具（例如 DS-5 流线分析器）来确定最适合于硬件加速的功能，并将最好在软件中实现的功能隔离起来。

3.2. 连接器件 I/O 与 HPS 外设和存储器的设计考量

配置 HPS 时最重要的考虑因素之一是了解如何管理 Cyclone V/ Arria V SoC 器件的 I/O。HPS I/O 在物理上分为：

- HPS Column I/O: 包含 HPS Dedicated Function Pins 和具有 loaner 功能的 HPS Dedicated I/O
- HPS Row I/O: 包含 HPS External Memory Interface (EMIF) I/O 和 HPS General Purpose Input (GPI) 管脚

图 2. Cyclone V SX 和 ST 器件中 HPS Column I/O 和 HPS Row I/O 的布局实例



注意: 有关 I/O 管脚布局的更多信息，请参阅 *Cyclone V* 或 *Arria V* 器件手册，第一卷：器件接口和集成中相应的“**I/O 功能**”章节。

表 6. HPS I/O 管脚类型摘要

管脚类型	用途
HPS Dedicated Function Pins	每个 I/O 仅有一个功能，不可用于其他用途。
HPS Dedicated I/O with loaner capability	这些 I/O 主要由 HPS 使用，如果 HPS 不使用这些管脚，则 FPGA 可以单独使用每个管脚。
HPS External Memory Interface (EMIF) I/O	这些 I/O 用于连接 HPS 外部存储器接口 (EMIF)。请参阅相应器件手册中的“ Cyclone V 器件中外部存储器接口 ”或“ Arria V 器件中外部存储器接口 ”章节了解有关这些 I/O 管脚布局的更多信息。
HPS General Purpose Input (GPI) Pins	这些管脚是 HLGPI 管脚。这些“仅输入”管脚与 HPS EMIF I/O 位于同一 bank 中。请注意，最小的 Cyclone V SoC 封装 U19 (484 管脚) 没有 HPS GPI 管脚。
FPGA I/O	通用 I/O，可用于 FPGA 逻辑和 FPGA External Memory Interfaces。

下表总结了每种 I/O 类型的特性。

表 7. I/O 类型

	HPS Dedicated Function Pins	HPS Dedicated I/O with loaner capability	HPS External Memory Interface	HPS General Purpose Input	FPGA I/O
可用 I/O 的个数	11	最多达到 67 个 (Cyclone V SoC) 和 94 个 (Arria V SoC)	最多达到 86 个	14 个 (Cyclone V SoC U19 封装除外)	最多达到 288 个 (Cyclone V SoC)，和最多
继续..					

	HPS Dedicated Function Pins	HPS Dedicated I/O with loaner capability	HPS External Memory Interface	HPS General Purpose Input	FPGA I/O
					达到 592 个 (Arria V SoC)
支持的电压	3.3V, 3.0V, 2.5V, 1.8V, 1.5V	3.3V, 3.0V, 2.5V, 1.8V, 1.5V	LVDS I/O 用于 DDR3, DDR2 和 LPDDR2 协议	与 HPS EMIF 使用的 I/O bank 电压相同	3.3V, 3.0V, 2.5V, 1.8V, 1.5V, 1.2V
用途	时钟, 复位, HPS JTAG	引导源, 高速 HPS 外设	连接到 SDRAM	通用输入	通用 I/O
时序约束	固定	固定	合法组合中是固定的	固定	用户定义
建议的外设	JTAG	QSPI, NANDx8, eMMC, SD/MMC, UART, USB, EMAC	DDR3, DDR2 和 LPDDR2 SDRAM	GPI	低速外设 (I ² C, SPI, EMAC-MII)

注意: 查看 [Cyclone V Device Datasheet](#) 或 [Arria V Device Datasheet](#) 中的 HPS 时序, 通过访问时序信息执行片外分析。

相关链接

- [Cyclone V 器件中的 I/O 特性](#)
Cyclone V 器件手册, 第一卷: 器件接口和集成中的章节
- [Arria V 器件中的 I/O 特性](#)
Arria V 器件手册, 第一卷: 器件接口和集成中的章节

3.2.1. HPS 管脚分配设计考量

由于 HPS 中包含的外围数量多于 HPS Dedicated I/O 可连接的总数, 因此 Platform Designer (Standard) 中的 HPS 组件提供管脚复用设置以及将大多数外设路由到 FPGA 架构的选项。同时, 在 HPS Dedicated I/O 未使用的管脚中, 任何具有接借贷 (loaner) 能力的管脚都可用作 FPGA 通用 I/O。

需注意的是 HPS I/O Bank 只能支持 1.2V, 1.35V, 1.5V, 1.8V, 2.5V, 3.0V 或 3.3V 电源的单电源, 具体取决于特定 bank 要求的 I/O 标准。仅 HPS Row I/O bank 支持 1.35V。

指南: 确保首先从 USB 开始将 USB, EMAC 和 Flash 接口路由到 HPS Dedicated I/O。

建议首先将高速接口 (例如 USB, Ethernet 和 flash) 路由到 HPS Dedicated I/O。必须将 USB 路由到 HPS Dedicated I/O, 因为它在 FPGA 架构中不可用。闪存引导源也必须路由到 HPS 专用 I/O (而非任何 FPGA I/O), 因为在配置 FPGA I/O 之前, 仅有这些 I/O 能够发挥作用。

注意: 由于可用的 HPS I/O 数量减少, 因此 Cyclone V SoC U19 封装 (484 管脚数) 仅能使用一个 USB 控制器 (而不是两个)。请参阅 Knowledge Base 中的 [Why can't I map USB0 to HPS IO in my Cyclone V SoC U19 package \(484 pin count\)?](#) 了解更多信息。

指南：在 **Platform Designer (Standard) HPS Component** 中使能 **HPS GPI** 管脚（如有需要）

默认情况下，未使能 Platform Designer (Standard) 中的 HPS GPI 接口。要使能该接口，必须在 Cyclone V/ Arria V 的 Platform Designer (Standard) HPS Component 中点选 “Enable HLGPI interface” 复选框。然后，这些接口会作为 Platform Designer (Standard) HPS Component Conduit Interface 的一部分显现，并且可以在设计顶层对他们进行单独分配。

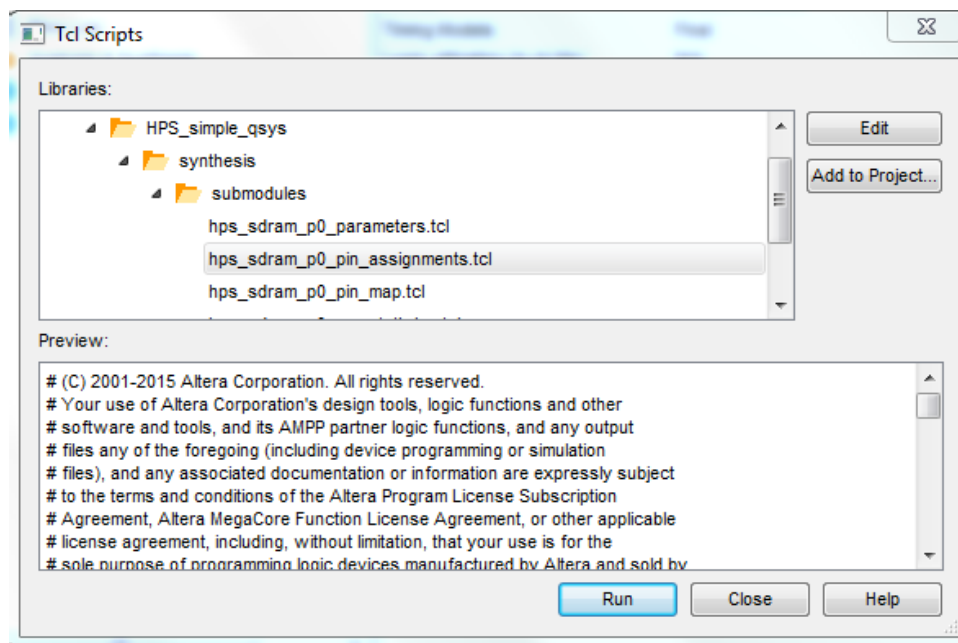
3.2.2. HPS I/O 设置：约束和驱动强度

指南：请确保具有 **HPS Dedicated I/O** 的 **I/O 设置**（驱动强度，I/O 标准，弱上拉使能等。）

生成包含 HPS 的 Platform Designer (Standard) 系统时，将自动管理 HPS 管脚位置分配。对于 HPS SDRAM，一旦运行 “hps_sdram_p0_pin_assignments.tcl” 脚本（该脚本在生成 Platform Designer (Standard) HPS Component 时创建），便完成了 I/O 标准和端接设置。

注意：

生成 Platform Designer (Standard) HPS Component 后，可在以下目录中找到脚本
“hps_sdram_p0_pin_assignments.tcl”：<Quartus project directory>
\<Platform Designer (Standard) file name>\synthesis\submodule。以下显示的是从 Intel Quartus Prime 中选择的脚本实例：



唯一必须管理的是针对 HPS Dedicated Function Pins 和 HPS Dedicated I/O 的 HPS I/O 约束。如同 FPGA 约束一样，诸如驱动强度，I/O 标准和弱上拉使能等约束都会添加到 Intel Quartus Prime 工程中，并在第二阶段引导加载器配置 I/O 过程中应用于引导中的 HPS。对于 FPGA I/O，I/O 约束将应用于 FPGA 配置文件。

注意：

上电过程中，引导闪存器件所需的 HPS Dedicated I/O 由 Boot ROM 配置，具体取决于 BSEL 值。

3.3. HPS 时钟和复位设计考量

HPS 子系统的主时钟和复位是 HPS_CLK1, HPS_CLK2, HPS_nPOR, HPS_nRST 和 HPS_PORSEL。HPS_CLK1 为 Main PLL 提供时钟源, 而 Main PLL 将生成 MPU, L3/L4 子系统, 调试子系统和 Flash 控制器的时钟。还可对其编程来驱动 Peripheral 和 SDRAM PLL。与此同时 HPS_CLK2 也可作为 Peripheral 和 SDRAM PLL 的时钟源。

HPS_nPOR 提供冷复位输入, 而 HPS_nRST 提供双向热复位源。对于 HPS_PORSEL, 首先它是一个输入管脚, 可用来为 HPS 块选择标准 POR 延迟或快速 POR 延迟。

注意: 请参阅 [Cyclone V Device Family Pin Connection Guidelines](#) 或 [Arria V G, GX, ST 和 SX Device Family Pin Connection Guidelines](#) 了解有关连接 HPS 时钟和复位管脚的更多信息。

3.3.1. HPS 时钟规划

指南: 使用 **Platform Designer (Standard)** 验证 MPU 和外设时钟

使用 Platform Designer (Standard) 来初始定义您的 HPS 组件配置。设置 HPS 输入时钟, 和外设源时钟和频率。请注意 Platform Designer (Standard) 的所有告警或错误消息。可以通过修改时钟设置来解决这些问题。某些情况下, 可能需要确定特定的告警条件不会影响您的应用程序。

3.3.2. 早期管脚规划与 I/O 约束分析

指导: 选择 HPS Dedicated Function I/O 的 I/O 电压水平

HPS_CLK1, HPS_CLK2, HPS_nPOR and HPS_nRST 由 VCCRSTCLK_HPS 供电。这些 HPS Dedicated Function Pin 为 3.3V, 3.0V, 2.5V 或 1.8V LVCMOS/LVTTL。这些管脚的 I/O 信号电压由 VCCRSTCLK_HPS 采用的电源电平确定。

注意: HPS_PORSEL 可连接到 VCCRSTCLK_HPS (用于快速 HPS POR 延迟) 或 GND (用于标准 HPS POR 延迟)。

注意: 如果 VCCIO_HPS 和 VCCPD_HPS 以及 VCCRSTCLK_HPS 共享相同的电压要求, 则他们可以共享相同的功率和稳压器。不需要在保持 HPS 运行的同时关闭 FPGA 架构的功能。

3.3.3. HPS JTAG, 时钟, 复位和 PoR 的管脚功能和连接

指南: 在使用 HPS (已供电) 的情况下, 请在 HPS_CLK1 上提供一个自由运行的时钟以访问 SoC device HPS JTAG。

要访问 HPS JTAG, 需要一个有效时钟源来驱动 HPS_CLK1。

指南: 以菊花链方式连接单个器件的 FPGA 和 HPS JTAG 时, 请确保 HPS JTAG 是该链中的第一个器件 (即, 位于 FPGA JTAG 之前)。

将 HPS JTAG 放置在 FPGA JTAG 前面, 可使 ARM DS-5 调试器启动对 HPS 的热复位。但是, 在冷复位情况下, 整个 JTAG 链都会断开, 直到冷复位完成为止, 该内容将在下一部分中讨论。

指南：考量电路板设计以隔离 HPS JTAG 接口

HPS Test Access Port (TAP) 控制器在冷复位时复位。如果 HPS JTAG 和 FPGA JTAG 被菊花链连接在一起，则整个 JTAG 链会断开，直到冷复位完成。如果 HPS 冷复位期间需要访问 JTAG 链，则在电路板设计中允许旁路 HPS JTAG。

指南：HPS_nRST 是一个开漏，双向专用热复位 I/O。

HPS_nRST 是低有效电平，开漏型双向 I/O。外部置位逻辑低至 HPS_nRST 管脚将启动 HPS 子系统热复位。也可从内部源（如软件启动的复位和从 FPGA 架构的复位请求）置位 HPS 热复位和冷复位。HPS 在内部置于热复位状态后，HPS 组件将成为复位源并将 HPS_nRST 管脚驱动到低电平，从而将所有连接的板级组件复位。

指南：遵守 HPS_nPOR 和 HPS_nRST 的最小置位时间规范。

复位 HPS_nPOR 上的信号，并且必须按照 Cyclone V 器件数据表或 Arria V 器件数据表中 HPS 部分规定 HPS_CLK1 周期的最小周期数置位 HPS_nRST 管脚。

3.3.4. 内部时钟

指南：避免在 HPS 和 FPGA 之间级联 PLL

FPGA 和 HPS 之间的级联 PLL 尚未表征。除非执行抖动分析，否则请勿将 FPGA 和 HPS PLL 链接在一起，因为无法保证从 FPGA 中最后一个 PLL 发出的是一个可靠的时钟。HPS 的输出时钟将不会馈入 FPGA 中的 PLL。

3.4. HPS EMIF 设计考量

HPS 子系统的关键组件是外部 SDRAM 存储器。Cyclone V 和 Arria V SoC 器件中，HPS 的专用 SDRAM Subsystem 对接 HPS External Memory Interface I/O。

查看以下指南，正确设计存储器和 HPS 之间的接口。该指导内容对于成功连接外部 SDRAM 与 HPS 至关重要。

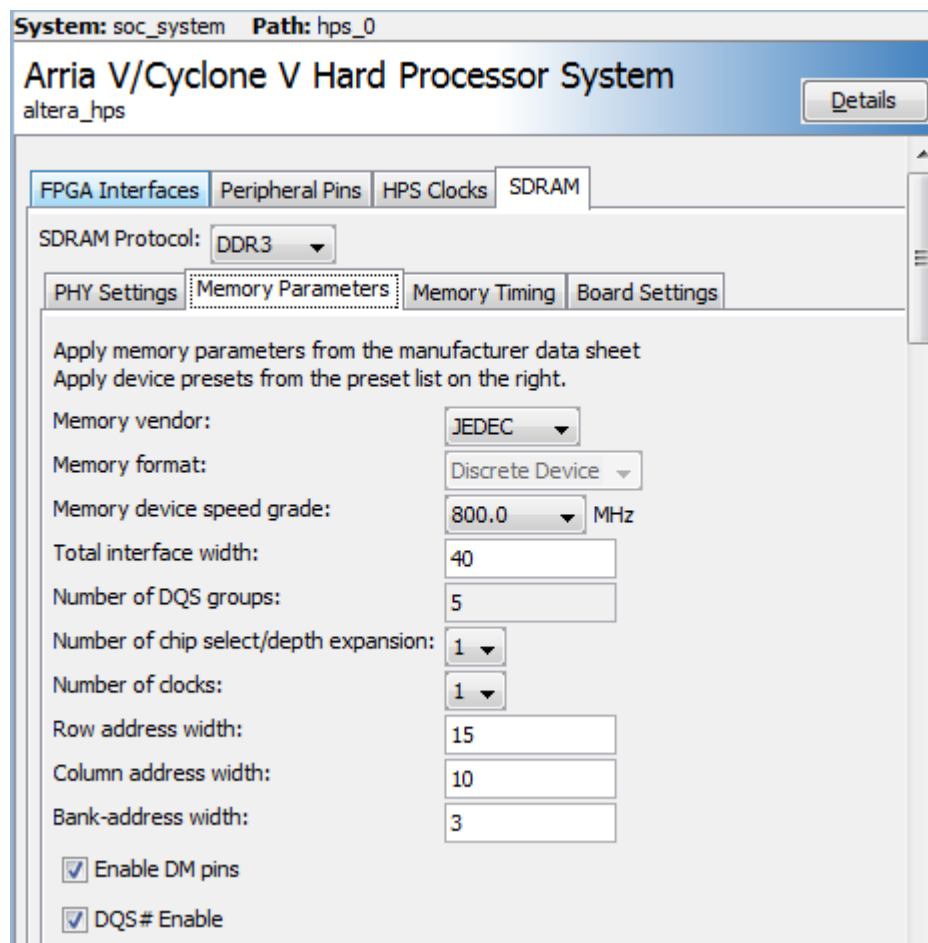
外部存储器接口手册，第三卷：参考资料 中包含 HPS 存储器控制器的功能描述。并列出 DDR3，DDR2 和 LPDDR2 支持的接口选项。

3.4.1. HPS 连接 SDRAM 的考量

指南：确保已使能 HPS 存储器控制器 Data Mask (DM) 管脚

Platform Designer (Standard) 的 HPS Component 中，请确保开启使能数据掩码管脚的复选框。如果未开启该控制，则只要主机访问小于存储器本地字的 SDRAM 数据，就随时发生数据损坏。

图 3. 在 HPS Component 中设置 Enable DM Pins 选项



确定您的 SDRAM Memory 类型和位宽度。Cyclone V 和 Arria V SoC 器件为 HPS 提供 DDR3, DDR2 和 LPDDR2 SDRAM 支持。

指南：对于特定的器件/封装组合，请确保配置中仅选择 **Cyclone V** 或 **Arria V HPS EMIF** 支持的 **DDR3**、**DDR2** 或 **LPDDR2** 组件或模块。

External Memory Interface Spec Estimator（可在 [External Memory Interface](#) 页面上找到）是一种参数化工具，可让您比较 Intel FPGA 和 SoC 器件中支持的外部存储器接口类型，配置和最大性能特征。

首先，过滤“Family”以仅选择 Cyclone V / Arria V SoC 器件。然后，继续对“Interface Type”使用过滤器以选择“HPS Hard Controller”

指南：请确保 **HPS Component** 中的 **Memory Clock Frequency** 是器件速度等级支持的时钟频率。

要获得器件速度等级支持的最大存储器时钟频率。请参阅 [External Memory Interface](#) 网页中的 External Memory Interface Spec Estimator 部分。

3.4.2. HPS SDRAM I/O 位置

Cyclone V 和 Arria V SoC HPS External Memory Interface I/O 位置固定，具体取决于所使用存储器的类型。可参阅“HMC Pin Assignment for DDR3/DDR2”和“HMC Pin Assignment for LPDDR2”下的器件 Pin Out 文件了解存储器接口管脚使用的确切 I/O 管脚。

注意: 未使用的 HPS External Memory Interface I/O Pins 不可分配给 HPS Peripherals，或由 FPGA 用作 Loaner IO。

注意: 相比较大封装（40-bit），最小的 Cyclone V SoC 封装 U19（484 管脚数）具有较窄的 HPS SDRAM 宽度（32-bit）。请参阅 *Cyclone V Device Handbook Volume 1: Device Interfaces and Integration* 中的“[External Memory Interfaces in Cyclone V Devices](#)”章节了解更多信息。

相关链接

文档: [Intel FPGA 器件的 Pin-Out（管脚说明）文件](#)

3.4.3. HPS EMIF 与 SoC FPGA 器件集成

将 Cyclone V 或 Arria V SoC HPS EMIF 与 SoC 系统设计的其余部分进行集成时，请考虑如下内容：

指南: [按照指导优化访问 HPS SDRAM 的所有主机的带宽](#)

通过 L3 Interconnect（FPGA-to-SDRAM 桥接除外）访问连接 HPS EMIF 的 SDRAM。在 FPGA 内核中设计和配置高带宽 DMA 主控以及相关缓冲器时，请参阅 [DMA 考量](#)（第 24 页）。上述部分涵盖的原则适用于所有高带宽 DMA 主控（例如，DMA 控制器组件，定制外设中的集成 DMA 控制器）和 FPGA 内核中通过 FPGA-to-SDRAM 和 FPGA-to-HPS 桥接端口访问 HPS 源（例如 HPS SDRAM）的相关缓冲，而非紧密耦合的 HPS 硬件加速器。

3.4.4. HPS 存储器调试

Cyclone V / Arria V HPS EMIF 不支持外部存储器接口工具包。要调试 HPS EMIF，可以更改预加载软件内的设置使能 Runtime Calibration Report 和 Debug Level 信息。此外，可使用预加载软件来检查 HPS SDRAM PLL 的状态。

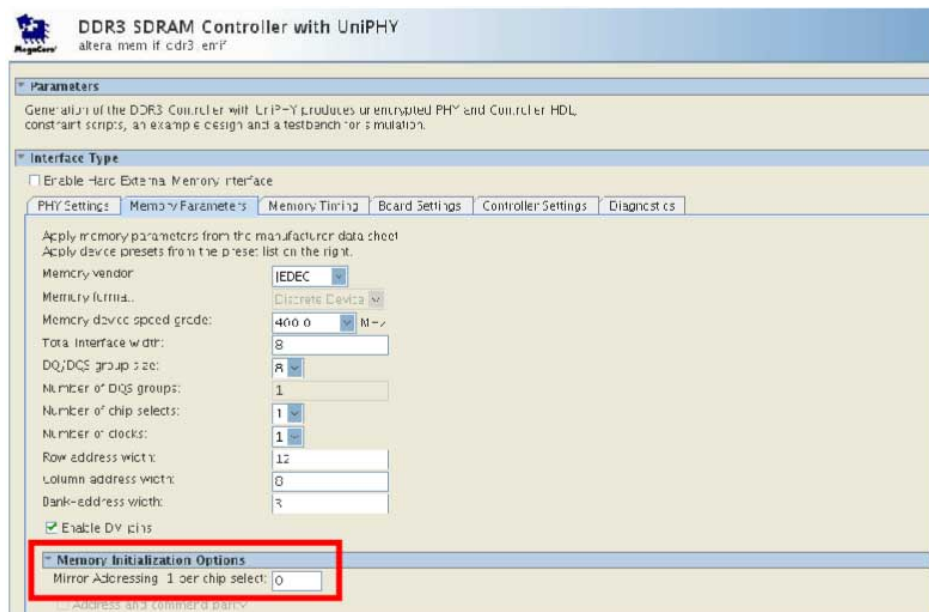
请参阅[使用 Preloader 调试 HPS SDRAM](#)（第 58 页）了解更多信息。

3.4.5. HPS 地址镜像技术

Cyclone V 和 Arria V HPS EMIF 支持地址镜像技术。在 **DDR3 SDRAM Controller with UniPHY GUI** 的 **Interface Type** 子窗口下找到 **Memory Parameters** 选项卡，从该选项卡中的 **Memory Initialization Options** 子窗口中开启该功能。例如，对于 4 个片选，输入 1011 镜像片选 #3, #1 和 #0 上的地址。

有关地址镜像的更多信息，请参阅[外部粗布存储器接口手册，第一卷：介绍和规范](#)。

图 4. HPS 地址镜像



相关链接

外部存储器接口手册，第一卷：介绍和规范

3.5. DMA 考量

3.5.1. 选择 DMA 控制器

选择最适合您设计的 DMA 实现

需要 DMA 来提高系统性能时，可使用集成到 HPS 或 FPGA 中软 DMA 模块中的 DMA。在决定选择要使用的选项时，应该考虑以下内容：

- HPS DMA：主要用于将数据移入和移出其他低速 HPS 模块，例如 SPI 和 I²C，以及对/从 HPS 存储器执行 memcop 功能。
- Soft DMAs：主要用于将数据移入或移出 FPGA 的外设。

3.5.2. 通过 HPS 互连优化 DMA 主控带宽

FPGA DMA 主控可通过 FPGA-to-HPS Bridge 和 FPGA-to-SDRAM Interface 访问 HPS 资源，并可在 HPS Platform Designer (Standard) Component 中进行配置。HPS SDRAM 控制器多端口前端 (MPFE) 提供这些资源的仲裁，并强制执行 Quality of Service (QoS) 设置。在规划和设计 DMA 主控和通过 HPS 互连访问资源的相关缓冲时，研究学习 HPS 互连的体系结构并考虑如下可用于通过互连优化带宽的指导和资源。

指南：利用 Cyclone V FPGA-to-HPS 桥接设计实例进行调整以获得较好的性能。

Cyclone V FPGA-to-HPS Bridge Design Example 是一个有用的平台。用于对 FPGA 和 HPS 资源之间特定数据流量访问模式进行建模。

本设计实例中包含一个实用工具，可使用该工具选择端点之间的数据通路，选择事务特性（例如，突发长度），并报告传输带宽。该使用工具在 HPS 中的 ARM Cortex* A-9 处理器上运行。

3.5.3. FPGA 加速器的时序收敛

面向 FPGA 开放的 HPS 桥接和 FPGA-to-SDRAM 接口是同步的，并且在本接口内运行时钟交叉。因此，仅需要确保 Timing Analyzer 中有面向 FPGA 的逻辑和中您的用户设计收敛时序。HPS 认为中断为异步，从而 HPS 逻辑将中断重新同步到内部 HPS 时钟，所以无需对他们进行时序收敛。

管道中有一些信号，这些信号不符合 Platform Designer (Standard) 支持的任何标准接口。具体实例表现为，路由到 FPGA 逻辑的 HPS 外设外部接口或 HPS DMA 外设请求接口。

3.6. 管理 FPGA 加速器一致性

可通过 HPS 或 FPGA 随时修改 HPS 和 FPGA 逻辑间的共享数据。许多应用程序需要数据一致性，意味着更改会传播至整个系统，以便每个主接口访问最新数据值。

针对数据一致性进行设计时，首先必须确定哪些数据传输需要保持一致。默认情况下，假定 FPGA 和 HPS 之间的所有访问都是不一致的，除非被软件明确管理的一致性或使用 HPS (SCU 和 ACP) 的一致性硬件功能管理一致性。

要确定 FPGA 中的外设是否需要一致访问 HPS 存储器，请回答以下问题：

- MPU 是否需要访问 FPGA 外设中生成的数据？
- FPGA 外设是否需要访问 MPU 生成的数据？

如果两个问题的答案都是“**Yes**”，则数据必须一致。可使用 ACP 让 FPGA 与 HPS 中的可高速缓存数据保持一致。

3.6.1. 高速缓存一致性

有几种机制的一致性是通过系统维护的：

HPS 在 MPU 子系统内的 1 级存储器子系统级上维护高速缓存一致性。MPU 子系统中内置的监听控制单元 (SCU) 使用修改后的专用共享无效 (MESI) 一致性协议维护两个 L1 数据闪存之间的高速缓存一致性。

3.6.2. FPGA 逻辑和 HPS 之间的一致性：加速器一致性端口 (ACP)

SCU 的加速器一致性端口 (ACP) 为系统中的其他主控 (包括 FPGA 架构中实现的逻辑) 提供运行高速缓存一致性访问的方法。为确保高速缓存一致性，仅可对 ACP 进行单项访问，这样就意味着此次访问中数据是最新的，但 SCU 并不负责随着时间的推移仍保持该数据的一致性。例如，如果 FPGA 中的主控从 ACP 读取到数据，然后处理器将存储器中的数据更新为相同数据，从而 FPGA 不在包含数据的最新数据副本。

3.6.3. 数据大小影响 ACP 性能

从使用 ACP 的加速器性能探索表明，随着 AXI 主控通过 ACP 端口传输的数据包尺寸的增加，提高了加速器性能，但只能达到一定程度。之后，就不再缓存整个数据包，且加速器性能下降。

指南：使用 ACP 管理小型数据访问的一致性，以及管理软件中大型数据的一致性。

3.6.4. 避免 ACP Dependency 锁定

ACP 和 CPU 的某些一致性访问过程中会导致死锁。但是，可通过简单的访问策略来避免这个类型的死锁。

如下实例中，由于 CPU 发起对 FPGA 逻辑中 HPS 的访问而导致死锁：

1. CPU 发起对 FPGA 逻辑中器件存储器的访问。CPU 流水必须停顿，直到完成该类型的访问。
2. 在 FPGA 逻辑状态机可以响应器件存储器访问之前，其对 HPS 的访问必须一致。发起一致性访问过程中需要 ACP。
3. 必须先执行 ACP 高速缓存维护操作，然后才能完成该访问。然而，CPU 的流水停顿会阻止执行高速缓存维护操作。从而导致死锁。

可以将所需访问拆分成较小的部分来实现，并且避免出现死锁。例如，可通过一个访问发起高速缓存维护操作，然后通过另一个访问确定操作状态。

3.6.5. FPGA 通过 AXI 或 Avalon-MM 对 ACP 的访问

AXI 协议支持主控发布可缓存的访问，但是 Avalon-MM 协议不支持该功能。对于运行可缓存访问的 FPGA 主控，该主控必须遵守 AXI 协议并能够执行可缓存的访问，还要将 ARCACHE[1] 或 AWCACHE[1] 设置为 1，ARUSER[0] 或 AWUSER[0] 设置为 1。

3.6.6. ACP 和 L2 Cache ECC 访问的数据对齐

L2 缓存以 64 位为一组执行错误检测和纠正，无需使用字节使能。

指南：对 ACP 的访问必须是 **64-bit** 对齐的，完整 **64-bit** 访问，并且不能在写入时禁用任何字节通路。

主 L3 开关和 ACP 端口都是 64 位宽，所以只需在调整大小后提供后 64-bit 对齐的，64 位宽高速缓存一致访问。

请求主控和 ACP 之间的 L3 互连中会出现数据大小调整。因此，如果该访问与 8 字节边界对齐并且主控执行大小为 2，4，8 或 16 字节的突发，则 32-bit 访问可与 L2 高速缓存 ECC 逻辑兼容。FPGA-to-HPS 桥接内也可能出现数据大小调整。

指南：确保 FPGA 访问符合 L2 cache ECC 要求的最简单方法是，在 FPGA 架构中实现 **64-bit** 主控并配置 **FPGA-to-HPS** 桥接以使 **64-bit** 从端口显现出来。这样就能确保不再需要重新调整 AXI 事务的大小。FPGA 中的逻辑还必须进行完整 **64-bit** 访问。

3.7. IP 调试工具

Intel Quartus Prime Design Software 包括许多用于 FPGA 硬件设计的 IP 和系统及调试工具。

以下工具常用于嵌入系统中的系统和 IP 调试：

- **Signal Tap** - 通过 FPGA 资源构建的片上逻辑分析仪
 - Avalon-MM v2 协议
 - AXI v3 协议
- **System console** - 基于服务的 API，用于控制软逻辑和将数据移入/移出 FPGA。

每个调试工具在硬件设计的不同阶段引入。典型硬件设计流程中，开发人员通常遵循以下高级验证步骤：

1. RTL 中的 IP Creation
2. IP 的 Testbench 和 BFM 验证
3. 在 IP 的硅芯测试中，使用系统控制台将激励驱动存储器映射或流接口
4. 在 IP 的硅芯测试中，使用低级别软件在 HPS 的处理器上运行

在 **Signal Tap** 和系统控制台中，如果两者都使用 FPGA JTAG 接口进行数据通信，则可以同时使用这两个功能。例如，可在 **Signal Tap** 中配备一个触发条件并通过 **System console** 控制的 JTAG-to-Avalon 桥接 IP 触发该条件发生。这些功能可与通过 JTAG 通信的 HPS 工具同时使用。

Cyclone V/ Arria V SoC 器件上有 2 个 JTAG 接口。其中第一个接口连接到器件的 FPGA 侧，而第二个接口连接到 HPS 调试访问端口 (DAP)。

相关链接

- [Intel Quartus Prime Standard Edition 手册第三卷“使用在系统源和探针的设计调试”章节中的：验证](#)
Signal Tap 说明
- [Avalon-MM v2 协议](#)
- [AXI v3 协议](#)
- [Intel Quartus Prime Standard Edition 手册第三卷“系统调试工具概述”章节：验证](#)
System Console 说明

4. SoC FPGA 的电路板设计指南

4.1. 电路板启动考量

本节介绍的考量内容有助于启动电路板。

4.1.1. 保留的 BSEL 设置

在启动的初始阶段，如果无法建立目标 JTAG 连接，可以将 BSEL 设置为 0x0 “Reserved” 来阻止 BootROM 从特定引导源引导，这样可能会有所帮助。然后可以下载测试程序并通过调试器运行测试。

4.2. 引导和配置设计考量

4.2.1. 引导设计考量

4.2.1.1. 引导源

指导：确定支持的引导源。

可从各种源引导 Cyclone V SoC / Arria V SoC 的 HPS 侧，具体由 BSEL 管脚选择：

- SD/MMC Flash
- QSPI Flash
- NAND Flash
- FPGA Fabric

每个可能的引导源都有其自身优势：

- SD 卡价格便宜，普遍可用并且存储容量大。工业版的可靠性更好。因为他们由 NAND flash 管理，所以在内部执行磨损均衡（wear leveling）和损坏块管理。
- eMMC 器件的封装较小但容量大，比 SD 更可靠。该器件不可拆卸，更有益于较为稳定地操作。
- QSPI 器件非常可靠，通常每扇区有至少 100,000 个擦除周期。但是，该器件的容量小于其他选择。他们通常被用作引导源，不会用作应用程序文件系统。
- 有较大尺寸的 NAND 器件可供选择，但他们并不由 NAND 管理，从而需要在软件中实施磨损均衡和损坏块管理等技术。
- FPGA 引导允许 HPS 在没有外部 Flash 器件的情况下进行引导。FPGA 引导存储器可综合 FPGA 资源（通常是预初始化嵌入式存储器块）或者可将存储器连接到 FPGA，例如外部 SRAM 或 SDRAM。要从 FPGA 引导，必须适用传统配置机制配置 FPGA。

4.2.1.2. 选择需要的 Flash 器件

指南：选择引导 flash 器件。

在选择 SoC FPGA 器件所使用的 flash 器件时，需重点考虑如下内容：

- **该 flash 器件是否与 HPS 引导 ROM 兼容？**：只可以从引导 ROM 支持的 flash 器件引导 HPS。
- **是否已验证该器件可以工作，并得到 Preloader, U-Boot 和 Linux 等软件支持？**：Intel 为支持的器件提供的是 Preloader, U-Boot 和 Linux 软件。而其他器件，则必须使用由用户开发的软件。
- **HPS Flash Programmer 是否支持该 Flash 器件？**：HPS Flash Programmer 允许使用 JTAG 连接写入 flash，主要用于对初始预 pre-loader/bootloader 映像进行编程。如果 HPS Programmer 不支持该器件，则可以使用其他 flash 编程方法，例如，使用 HPS 编程 flash，或者还可以使用 U-Boot 的 flash 编程功能。

请参阅 [Supported Flash Devices for Cyclone V SoC and Arria V SoC](#) 了解更多信息。

4.2.1.3. BSEL 选项

指南：配置选定引导源的 BSEL 管脚。

通过 BSEL 管脚选择引导源。

或许更改用于调试的引导源会有帮助，即使电路板没有其他可用的引导源。例如，一个从 QSPI 引导的电路板中，可选择保留的引导源，这样 BootROM 就不会工作，进而受益。或者选择从 FPGA 引导并在 FPGA 逻辑中放入一个测试映像。

如果系统允许（空间约束等原因），可规划提供交换机或至少提供电阻器，以便能够根据需要更改 BSEL。

4.2.1.4. 引导时钟

指南：确定引导时钟源。

引导时钟映像引导持续时间。在选择引导时钟时，首先考虑的因素是系统引导速度要求。系统引导时间要求取决于需要多快完成 FPGA 配置才能获得正确的响应时间，以及引导 HPS 软件必须达到的速度。HPS 软件引导受如下因素影响：

- HPS 外部时钟的值（如，OSC1 时钟）
- Boot flash 源接口操作频率

通过 CSEL 管脚选择引导时钟配置。相应的 *Hard Processor System Technical Reference Manual* 中有关于可用组合的说明。

注意：从 FPGA 逻辑引导时不使用 CSEL 管脚。

相关链接

- [Cyclone V 硬核处理器系统技术参考手册](#)
- [Arria V 硬核处理器系统技术参考手册](#)

4.2.1.5. CSEL 选项

指南：提供配置 **CSEL** 选项的方法。

出于调试用途时，即使终端产品仅需要一个 **CSEL** 设置，也可以允许设置多个 **CSEL** 值。如果可能，在设计电路板时也可采取这种方法，这样即使最终仅使用一个值，但是 **CSEL** 配置可能各不相同。这种可配置性或许对调试很有用，可以通过电阻器、跳线和交换机来完成。

4.2.1.6. 选择 NAND Flash Devices

指南：选择符合 **ONFI 1.0** 的 **NAND flash**。

从 **NAND** 引导时，请确保所选器件符合 **ONFI 1.0**。

用于引导的 **NAND** 器件还必须有一个 **x8** 接口，并且仅有一对 **ce#** 和 **rb#** 管脚。

尽管有些不符合 **ONFI 1.0** 的器件也与 **BootROM** 兼容，但 **HPS Flash Programmer** 仅支持 **ONFI** 合规器件。

4.2.1.7. 确定 Flash 编程方法

指南：请确保正确配置电路板以支持 **flash** 编程。

HPS Flash Programmer 是随 **SoC EDS** 提供的工具，可用于对 **Cyclone V / Arria V SoC** 板上的 **QSPI** 和 **NAND flash** 器件编程。该工具旨在写入相对少量的数据（例如 **preloader**），因为其通过 **JTAG** 工作并且速度有限。

如果使用 **HPS Flash Programmer** 工具，请确认其支持您计划使用的器件。[Intel SoC FPGA 嵌入式开发套件用户指南](#) 中罗列了支持的器件。

其他编程 **flash** 器件的方法是：

- 使用调试器编程 **Flash**（例如，**DS-5**）
- 从 **U-Boot** 编程 **Flash**
- 从 **Linux**（或其他 **OS**）控制台编程 **Flash**
- 通过专用硬件编程 **Flash**

相关链接

[Intel® SoC FPGA 嵌入式开发套件用户指南](#)

4.2.1.8. 为 QSPI and SD/MMC/eMMC 提供 Flash Memory 复位

指南：请确保 **QSPI** 和 **SD/MMC/eMMC** 器件具有在 **HPS** 复位后而随之复位的机制。

QSPI 和 **SD/MMC/eMMC flash** 器件可能会被软件置于一种状态，导致 **BootROM** 无法成功访问器件，这样可能会触发下次复位中的引导失败。出现该问题是因为 **HPS** 已复位，但是 **flash** 部分还未复位。

因此每次 **HPS** 复位（无论热复位或冷复位）时都需要复位 **QSPI** 和 **SD/MMC/eMMC** 引导 **flash** 器件。

请注意有些器件没有复位管脚。这种情况下，就必须通过其他方法重新启动 flash，例如使用 MOSFET。需要注意要求的最小复位脉冲持续时间。

4.2.1.9. 选择 QSPI Flash 器件

指南：对于逻辑应用程序，请避免使用大于 16 MB 的 QSPI flash 器件

16 MB 或更小的 QSPI flash 器件始终支持 3 字节寻址。因此 HPS Boot ROM 可对他们进行访问。使用此类器件时，即使 HPS 进行冷复位或热复位，都不需要复位或重新启动 flash。

指南：使用大于 16 MB 的 QSPI 器件时，如果器件支持，则请使用 QSPI 扩展 4-byte 寻址命令。

一些 QSPI flash 器件支持扩展命令集，从而允许主接口使用 4-byte 寻址但无需切换成 4-byte 寻址模式。如果使用扩展命令集，可以保留 flash 器件的 3-byte 寻址模式，以使 Boot ROM 可在下一个复位周期时继续对其进行访问，而无需复位或重新启动 flash 器件。

有关从 QSPI 引导的详细信息，请参阅 RocketBoards.org 网站中的 **CV SoC and AV Soc QSPI Boot** 部分。

4.2.2. 配置

Cyclone V / Arria V SoC 器件支持两种主要类型的 sof 配置流程：

- 传统 FPGA 配置
- 启用 HPS 的 FPGA 配置

已开启 HPS 的配置使用快速被动并行（FPP）模式，允许 HPS 使用其可访问的存储位置（例如，QSPI, SD/MMC 和 NAND flash）配置 FPGA。Cyclone V/ Arria V SoC 的 FPGA 配置流程与 Cyclone V/ Arria V FPGA 器件相同，其中的外部配置数据源连接 FPGA 中的控制模块。

4.2.2.1. 传统配置

传统的 FPGA 配置流程中 FPGA 由外部源（例如 JTAG，有效串行或快速被动并行）进行配置。

4.2.2.2. 启用 HPS 的配置

器件上电后 HPS 开始执行引导 ROM 中的软件，所有器件 I/O 默认为输入三态操作模式。引导 ROM 根据采样 BSEL 管脚配置专用引导 I/O。

4.2.3. 参考资料

请参阅如下参考资料了解更多信息。

相关链接

- [Intel® SoC FPGA 嵌入式开发套件用户指南](#)
- [Support tab on Cyclone V SoCs](#)
Cyclone V SoC 文档
- [Support tab on Arria V SoCs](#)
Arria V SoC 文档
- [SoC 嵌入式开发套件入门指南](#)

- [Golden System Reference Design \(GSRD\) User Manuals](#)
- [AN-709: HPS SoC 引导指南- Cyclone V SoC 开发工具](#)

4.3. HPS 电源设计考量

有关功耗和热分析，SoC 器件管脚链接，电源设计和去耦的设计考量与建议，请参阅 [Arria V](#) 和 [Cyclone V](#) 设计指南。

以下小节是关于 SoC 器件的补充内容。

相关链接

[Arria V](#) 和 [Cyclone V](#) 设计指南

4.3.1. 早期系统和电路板规划

4.3.1.1. 早期功耗估算

按照 **Early Power Estimation** 网站中的指导使用 Intel Quartus Prime 软件功率分析仪电子表格。

此外，使用 EPE 电子表格时请参考以下 [Cyclone V](#) 和 [Arria V](#) SoC 器件指导。

指南：使用 Main 工作表选择 Power Characteristics 设置的 “Maximum” 值。

在估算功耗以便设计的电源能够满足工艺，电压和温度（PVT）要求的最大功耗时，请采用器件的最大功耗特性。

指南：使用 I/O 工作表添加分配给 FPGA I/O 的 HPS 外设。

可在该选项卡中说明应用程序中 I/O Elements (IOEs) 的各种配置。使用 IO-IP 选项卡说明每个 I/O 设置背后的控制器 IP。

分配用于 FPGA I/O 的 HPS 外设中，请根据需要在电子表格中添加行来说明设计中各个 HPS 外设 I/O 的特性。

指南：使用 HPS 工作表选择每个 CPU 的 Frequency, Application 以及适用的 Application Mode。

每个 CPU 的 Application/Application Mode 设置支持从行业标准基准列表中进行选择，以模拟应用程序中的 CPU 利用情况。也可以选择 “Custom” 为 ALUs 和 cache 存储器定义一组唯一的 CPU 利用率参数。

指南：使用 HPS 工作表更新 HPS SDRAM Type, Frequency 和 Width。

请注意 SDRAM 类型的选择还会将 Bank 6A I/O 电压更新至 6B。

指南：使用 HPS 工作表更新 HPS I/O Bank Voltage 和 Peripheral Usage

从下拉菜单选择外设电压之前，请先确保至少有一个 HPS I/O 已配置到相同电压。

指南：对于具有 L 电源选项的 Cyclone V SoC，请在计算总静态功率时应用适当的乘法系数。

由于 Cyclone V 的 EPE 中没有 L-power 选项，因此需要使用如下步骤计算总静态功率：

1. 以标准（非-L）部件号键入设计和资源利用率。
2. 一旦获得总静态功率后，请应用乘法系数 0.7（25K-LE 和 40K-LE 器件）或 0.8（85K-LE 和 110K-LE 器件）计算得到较低静态功率，以用于具有 L-power 选项的器件。

注意: 乘法系数仅应用于总静态功率计算不适用于动态功率。

相关链接

[Early Power Estimators \(EPE\)](#) 和 [Power Analyzer](#)

4.3.2. SoC FPGA 器件 HPS 和 FPGA Power Supplies 的设计考量

4.3.2.1. 考虑需要在保持 HPS 运行的同时关闭 FPGA 部分

指南: 对 **FPGA** 使用单独的可编程稳压器，以支持在保持 **HPS** 运行时关闭 **FPGA**

Cyclone V/ Arria V SoC 器件能够在保持 HPS 运行的同时关闭 FPGA。为此，FPGA V_{CC} 必须由支持控制接口（例如 I²C）的可编程稳压器提供电源。比如开发板上的 Cyclone V SoC Development Kit 就支持该功能。可在 [Cyclone V SoC Development Kit and Intel SoC FPGA Embedded Development Suite](#) 中找到有关 Cyclone V SoC Development Kit 的信息。

请参阅 [Cyclone V SoC 智能配置](#) 设计实例了解如何使用 HPS 的 I²C 连接控制 FPGA 电源稳压器。

4.3.2.2. 考虑需要的 HPS 引导时钟频率

Cyclone V / Arria V SoC 器件支持 PLL 旁路模式下的 10-50 MHz HPS 引导时钟，在 PLL Locked 模式下最高可以达到 400MHz。上电或冷复位过程中，引导 ROM 对 CSEL 管脚的值进行采样，并在需要时配置 HPS PLL 以提供更快的引导时钟频率。

请参阅关于 CSEL 选项的表格，以及相应 *Hard Processor System Technical Reference Manual* 中“引导和配置”附录下对应的外部振荡器频率。

相关链接

- [Cyclone V Hard Processor System Technical Reference Manual](#) 的“引导和配置”附录
- [Arria V Hard Processor System Technical Reference Manual](#) 的“引导和配置”附录

4.3.3. 电路板设计的管脚连接考量

4.3.3.1. 器件上电

上电和断电排序

Cyclone V/ Arria V SoC 器件中需要考虑以下电源轨的上电顺序。

- VCC_HPS
- VCCPD_HPS
- VCCIO_HPS

- VCCRSTCLK_HPS
- VCCPLL_HPS
- VCC_AUX_SHARED

请参阅 *Cyclone V* 或 *Arria V* 器件手册第一卷：器件接口和集成中“电源管理”章节的“上电顺序”。

指南：设计 Power Distribution Network (PDN) 时，请考虑电源上最大瞬时电流的斜坡时间。

使用 PDN Tool 计算内核架构 VCC 电源所需的应用程序 PDN 目标阻抗时，请采用 Core Clock Frequency 和 Current Ramp Up Period 参数将 VCC 上的最大瞬时电流斜坡时间建模。相对于默认阶跃函数分析，此过程放宽了阻抗的要求，从而产生具有更少去耦电容器但更高效的 PDN。

可从 EPE Spreadsheet 中获得初始瞬时电流估算值，并且可在设计即将完成时使用 Intel Quartus Prime 中的 Power Analyzer Power Analysis Tool 进行更准确的分析。

请参阅 [AN 750：使用 Altera PDN 工具优化 供电网络设计](#)。

相关链接

- [Arria V 和 Cyclone V 设计指导](#)
- [Cyclone V 器件手册](#)
- [Arria V 器件手册](#)

4.3.4. 功率分析和优化

请遵循 [Arria V](#) 和 [Cyclone V](#) 设计指南 中 Power Analysis and Optimization 部分的指导。此外，请考虑将以下选项用于器件的 HPS 部分。

处理器和存储器时钟速度

对 HPS 中功耗影响最大的是处理器时钟速度和外部 SDRAM 编程存储器的类型，大小和速度。谨慎选择这些系统参数以满足应用程序的功能和性能要求，有助于最大限度降低系统功耗。

CPU 待机模式和动态时钟门控

可在整个 MPU 子系统中使用 CPU 待机模式和动态时钟门控逻辑。每个 CPU 都可置于待机模式，Wait for Interrupt 或 Wait for Event 模式，以进一步降低功耗。

有关待机的更多信息，请参阅 [Cortex-A9 Technical Reference Manual \(revision r2p0\)](#)。可从 [Design Examples](#) 网页上获得 Power Optimization Example。

管理外设电源

在 Platform Designer (Standard) 中配置 HPS 组件时，仅使能应用程序中使用的外设。配置外设获得最低时钟速度的同时保持功能和性能要求。通过将不活跃外设置于复位状态并关闭其时钟源就可节省软件控制下的其他功耗。

通过关闭电源开管理功耗

Cyclone V SoC 和 Arria V SoC 支持关闭器件中 FPGA 部分的电源，同时保持 HPS 运行。请参阅 [Cyclone V SoC Smart Configuration](#) 设计实例了解如何使用 HPS 中的 I²C 连接控制 FPGA 电源稳压器。

4.4. HPS 的边界扫描

指南：请确保上电 **HPS** 并将其保持在复位状态以后才执行对 **FPGA** 和 **HPS I/O** 的边界扫描测试。

HPS JTAG 不支持边界扫描测试 (BST)。要在 HPS I/O 管脚上执行边界扫描测试，必须使用 FPGA JTAG。

4.5. HPS 接口的设计指南

本小节概述 HPS 接口（如，EMAC PHY, USB, QSPI, SD/MMC, NAND Flash, UART, I²C 和 SPI）的设计指南。

4.5.1. HPS EMAC PHY 接口

在 Platform Designer (Standard) 内配置 EMAC 外设的 HPS 组件时，必须从以下支持的 PHY 接口中为每个 EMAC 实例选择一个接口：

- 使用专用 I/O 的 Reduced Gigabit Media Independent Interface (RGMII)
- 对接 FPGA 逻辑的 Media Independent Interface (MII)
- 对接 FPGA 逻辑的 Gigabit Media Independent Interface (GMII)

多个 HPS EMAC 实例中都可配置所支持 PHY 接口类型的任意组合。

指南：对使用 **HPS Dedicated I/O** 的 **RGMII** 情况下，开发一个早期 **I/O** 平面规划模板设计，可确保除了规划用于 **HPS Dedicated I/O** 的 **HPS** 外设以外，还有其他足够的 **HPS Dedicated I/O** 来对应所选的 **PHY** 接口。

注意：有关配置 HPS 组件的指导，请参阅 *Cyclone V* 或 *Arria V* 硬核处理器系统技术参考手册中的“HPS 组件介绍”章节。

可通过使用 HPS 组件使面向 FPGA 架构的 MII/GMII PHY 接口执行其他 PHY 接口标准，例如 RMII, RGMII, SGMII, MII 和 GMII—通过使用 FPGA 中的软适配逻辑，以及通用 FPGA I/O 和收发器 FPGA I/O 中的功能。

指南：选择 **PHY** 器件时，请考虑需要的 **Ethernet** 速率，可用的 **I/O** 和可用的收发器；提供偏移控制的 **PHY** 器件；以及器件驱动器可用性。

注意：请先了解可用于您所选操作系统的器件驱动程序或随 Cyclone V/ Arria V SoC 开发套件（黄金系统参考设计）提供的 Linux 器件驱动程序

Cyclone V/ Arria V SoC Hard Processor System (HPS) 能够使用 HPS 专用 I/O 管脚在任何支持的 I/O 电压下将其嵌入式 Ethernet MAC (EMAC) PHY 接口直接连接行业标准 Gigabit Ethernet PHY。这些电压通常包括 1.8V, 2.5V 和 3.0V。如果 HPS Dedicated I/O 管脚用于 PHY 接口，则不使用 FPGA 布线资源并且时序是固定的，从而简化了接口上的时序。本文档介绍说明最典型接口 RGMII 的设计指导。

还可通过 FPGA 架构将 HPS 连接到 HPS EMAC，其中 GMII 总线接口用于 Gigabit 访问，以及 MII 总线接口用于 10/100 Mbps 访问。GMII-to-SGMII 适配器也可用于自动适配基于收发器的 SGMII 光模块。

注意: 由于 Cyclone V/ Arria V SoC 器件中的一个勘误，当通过 HPS Dedicated I/O 路由时不支持 RMII PHY 接口。但是通过 FPGA 架构布线时，支持 RMII 接口。

相关链接

- [Cyclone V 硬核处理器系统技术参考手册](#)中的“HPS 组件介绍”章节
- [Arria V 硬核处理器系统技术参考手册](#)中的“HPS 组件介绍”章节
- [黄金系统参考设计 \(GSRD\) 用户手册](#)

4.5.1.1. 通过 HPS Dedicated I/O 连接的 PHY 接口

本小节讨论通过 HPS Dedicated I/O 的 RGMII PHY 接口的设计考量。

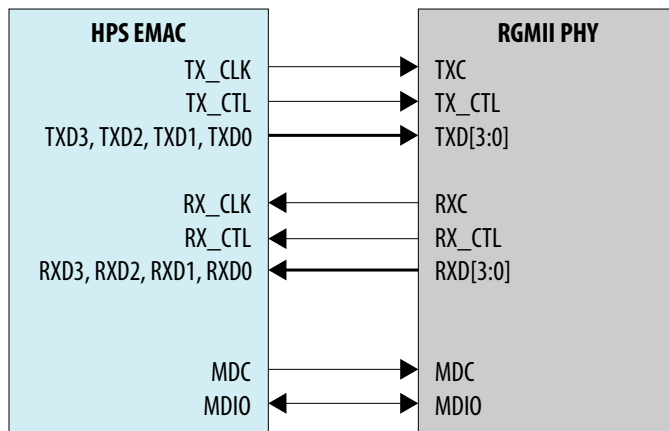
4.5.1.1.1. RGMII

Reduced Gigabit Media Independent Interface (RGMII) (Reduced GMII)是最常见的接口，因为它支持 PHY 层的 10 Mbps，100 Mbps 和 1000 Mbps 连接速度。RGMII 使用 4-bit 宽发送和接收数据通路，每个数据通路都有自己的源同步时钟。所有发送数据和控制信号源同步到 TX_CLK，与此同时所有接收数据和控制信号源同步到 RX_CLK。

所有速度模式中，TX_CLK 始终由 MAC 提供时钟源，而 RX_CLK 始终由 PHY 提供时钟源。在 1000 Mbps 模式下，TX_CLK 和 RX_CLK 为 125 MHz，用 Dual Data Rate (DDR)信令。

10 Mbps 和 100 Mbps 模式下，TX_CLK 和 RX_CLK 分别为 2.5 MHz 和 25 MHz，使用上升沿 Single Data Rate (SDR)信令。

图 5. RGMII



I/O 管脚时序

本小节从满足 1000 Mbps 模式下各种要求的角度触发，解决 RGMII 接口时序相关问题。1000 Mbps 模式下，需要最多接口时序裕量，因而这是此处唯一需要考虑的情况。

125 MHz 时，周期为 8 ns，但由于两个沿都已使用，因而有效沿周期仅为 4 ns。TX 和 RX 总线完全独立但时钟源同步，从而简化了时序。RGMII 规范要求 CLK 在任意方向上从接收器的 DATA 延迟最短 1.0 ns 到最长 2.6 ns。

换言之，MAC 到 PHY 的 TX_CLK 必须延迟于输出到 PHY 输入和从 PHY 输出到 MAC 输入的 RX_CLK。如在输出管脚处测得的那样，信号在每个方向的 +/- 500 ps RGMII 偏斜规范内同步传输。每个方向所需的最小延迟为 1 ns，但建议将目标延迟定为 1.5 ns 到 2 ns，以确保足够的时序裕量。

发送路径建立/保持

仅 TX_CLK 到 TX_CTL 和 TXD[3:0] 的建立和保持时间会影响发送。Cyclone V/ Arria V HPS Dedicated I/O 没有可编程延迟功能。

Cyclone V/ Arria V SoC 中的 TX_CLK，必须采用 RGMII 规范下的 1.0 ns PHY 最小输入建立时间。强烈建议将该延迟增加到 1.5 ns 至 2.0 ns。许多 PHY 提供可编程偏斜，与此同时有些支持 RGMII 2.0 规范的还默认使能发送和接收数据通路上的偏斜。

PHY 延迟和 FPGA I/O 延迟功能之间，必须确保 CLK 与 CTL 和 D[3:0] 之间的延迟 2 ns，或者大多数 PHY 的典型最小建立偏斜为 1.2 ns。请参阅您 PHY 供应商提供的数据表了解更多详细信息。

指南：请确保您的设计中包含必要的 **Quartus** 设置以针对所需延迟配置 **HPS EMAC** 输出。

在 Cyclone V/ Arria V SoC Development Kit 和相关 Golden Hardware Reference Design (GHRD 是 GSRD 的硬件组件) 中，通过 Microchip* (Micrel*) KSZ9021RN PHY 实现 PHY 偏斜。请参阅 hps_common_board_info.xml 文件和 Golden System Reference Design (GSRD) 中的 PHY 驱动编码。

接收路径建立/保持

对于接收时序，只需考虑 RX_CLK 到 RX_CTL 和 RXD[3:0] 的建立和保持时间。Cyclone V/ Arria V SoC HPS Dedicated I/O 中，PHY 侧或电路板走线延迟方面不需要其他考量。

指南：硬件开发人员应该指定需要的 **FPGA** 偏斜，以便软件开发人员可将偏斜添加到器件驱动编码。

使用 hps_common_board_info.xml 文件编译 Cyclone V 或 Arria V SoC GSRD 的 Linux 器件树。

4.5.1.2. 通过 FPGA I/O 连接的 PHY 接口

没有足够 HPS Dedicated I/O 容纳 PHY 接口时，或者 HPS EMAC 不支持需要使用的 PHY 接口时，将 FPGA I/O 用做 HPS EMAC PHY 接口会有所帮助。

指南：配置 **Platform Designer (Standard)** 内 HPS 组件时，请指定 **PHY** 接口发送时钟频率。

对于 GMII 或 MII（包括使用其他 PHY 接口），请指定 HPS EMAC PHY 接口的最大发送路径时钟频率：GMII 为 125 MHz，MII 为 25 MHz。该配置可确保 Platform Designer (Standard) 系统生成后，PHY 接口发送时钟应用正确的时钟时序约束。

相关链接

- [嵌入式外设 IP 用户指南](#)
- [Cyclone V RGMII 设计实例](#)

4.5.1.2.1. GMII/MII

仅 Cyclone V/ Arria V SoC 中有 MII 和 GMII，通过将 EMAC 信号驱动到 FPGA 内核布线逻辑，然后最终到 FPGA I/O 管脚或 FPGA 内核中的内部寄存器。

指南：应用时序约束并使用 Timing Analyzer 验证时序。

由于 FPGA 内核和 I/O 结构中的布线延迟各不相同，因此阅读时序报告很重要，尤其是对于 GMII，需要创建时序约束。GMII 有 1 个 125 MHz 时钟，并且不同于 RGMII，其属于单一数据速率。使用 GMII 时在 CLK-to-DATA 偏斜方面的考量上也不同；它的信号由设计自动居中，在下降沿发起并在上升沿捕获。

指南：FPGA I/O 边界处的寄存器接口 I/O。

由于内核和 I/O 延迟极易超出 8 ns，所以建议将这些总线寄存到 I/O Element (IOE)寄存器的每个方向，以便他们在内核 FPGA 逻辑架构中传输时仍保持对齐。发送数据和控制中，将信号锁存于 HPS EMAC 的 `emac[0,1,2]_gtx_clk` 输出的下降沿，来保持 clock-to-data/control 关系。将接收数据和控制锁存于 RX_CLK 信号上升沿的 FPGA I/O 输入中，该信号由 PHY 提供时钟源。

指南：考虑 MII 模式下的发送时序。

PHY 处于 100 Mbps 模式时，MII 为 25 MHz；PHY 处于 10 Mbps 模式时，MII 为 2.5 MHz，因此最短时钟周期是 40 ns。PHY 为发送和接收方向提供时钟源。发送时序是相对于 PHY 提供的 TX_CLK 时钟而言，可能要考虑转向时间，但因为 40 ns 时钟周期较长，通常并不成问题。

注意：事务通过 FPGA 路由，然后输出数据。往返延迟必须小于 25 ns，因为输入建立的时间为 15 ns。由 PHY 提供时钟源的 TX_CLK 信号中，其负边沿上的 HPS EMAC 发送路径逻辑将发送数据和控制驱动到 FPGA 逻辑中，这样可从 40 ns clock-to-setup 时序预算中消除 20 ns。

数据到达时序上的往返时钟路径延迟会导致 PHY-to-SoC 板传播延迟，再加上从 SoC 管脚到或通过 HPS EMAC 发送时钟多路复用的内部路径延迟会占用其余 20 ns 设置时序预算，或许需要对 FPGA 逻辑中 `phy_txclk_o` 时钟输出寄存器上升沿的发送数据和控制重新计时，以用作 MII 模式发送数据和控制。

4.5.1.2.2. 适配 RGMII

可使用 FPGA 中的逻辑将 GMII HPS EMAC PHY 信号适配到 FPGA I/O 管脚处的 RGMII PHY 接口。虽然可以按照适配情况来设计自定义逻辑，但本小节说明使用 Platform Designer (Standard)适配器 IP。

指南：使用 Platform Designer (Standard)中的 GMII-to-RGMII Adapter IP。

在 Platform Designer (Standard)中配置 HPS 组件，以用作 EMAC 的“FPGA” I/O 接口。请勿将生成的 HPS 组件 GMII 信号导入 Platform Designer (Standard)。反而需要将 Intel HPS GMII to RGMII Converter 添加到 Platform Designer (Standard)子系统，并连接 HPS 组件的 GMII 信号。GMII to RGMII Converter 使用 Platform Designer (Standard)中的 Intel HPS EMAC Interface Splitter 将 `emac` 管线从 HPS 组件分离，以供 GMII to RGMII Converter 使用。请参阅 [Embedded Peripherals IP User Guide](#) 了解如何使用 Intel HPS GMII to RGMII Converter。

指南：为 10/100 Mbps 模式提供无干扰时钟源。

RGMII PHY 接口中，TX_CLK 始终由 MAC 提供时钟源，但是在 10/100 Mbps 模式下 HPS 组件的 GMII 接口需要由 PHY 器件提供 TX_CLK。GMII to RGMII 适配逻辑必须在 GMII 的 `emac[0,1]_tx_clk_in` 输入端口提供 2.5/25 MHz TX_CLK，2.5 MHz 和 25 MHz 之间的切换必须按照 HPS EMAC 的要求，以无干扰方式完成。可使用 FPGA PLL 来提供 2.5 MHz 和 25 MHz TX_CLK 以及一个 ALTCLKCTRL 块，可在无干扰计数器输出之间进行选择。

注意： 请参阅 [Cyclone V RGMII Example Design](#) 了解本实现的硬件和软件实例。

4.5.1.2.3. 适配到 RMII

可使用 FPGA 中的逻辑将 MII HPS EMAC PHY 信号适配到 FPGA I/O 管脚处的 RMII PHY 接口。

指南：提供 50MHz REF_CLK 时钟源。

RMII PHY 将单个 50 MHz 参考时钟 (REF_CLK) 用于发送与接收数据和控制。通过板级时钟源，FPGA 逻辑中生成的时钟，或从可生成 REF_CLK 的 PHY 提供 50 MHz REF_CLK。

指南：适配发送和接收数据和控制路径。

FPGA 逻辑中公开的 HPS EMAC PHY 接口是 MII，10 Mbps 和 100 Mbps 操作模式下，分别需要 2.5 MHz 和 25 MHz 发送和接收时钟输入。发送和接收数据通路均为 4-bits 宽。RMII PHY 将 50 MHz REF_CLK 用于其发送和接收数据通路，以及 10 Mbps 和 100 Mbps 操作模式。RMII 发送和接收数据通路均为 2-bits 宽。10 Mbps 时，发送和接收数据和控制 在 50 MHz REF_CLK 的 10 个时钟周期内保持稳定。FPGA 逻辑中必须提供在 HPS EMAC MII 和外部 RMII PHY 接口之间进行适配的适配逻辑：4-bits @ 25 MHz/2.5 MHz to/from 2-bits@ 50 MHz，10 模式下 10x 过采样。

指南：HPS EMAC MII tx_clk_in 时钟输入上提供无干扰时钟源。

HPS 组件的 MII 接口要求其 `emac[0,1,2]_tx_clk_in` 输入端口上有 2.5/25 MHz 发送时钟，并且按照 HPS EMAC 的要求，完成 2.5 MHz 和 25 MHz 之间的切换必须无干扰。可使用 FPGA PLL 来提供 2.5 MHz 和 25 MHz 发送时钟，以及 ALTCLKCTRL 块以在无干扰计数器输出之间进行选择。

相关链接

[嵌入式外设 IP 用户指南](#)

4.5.1.2.4. 适配到 SGMII

可使用 FPGA 和高速串行收发器 (MGT) I/O 中的逻辑将 GMII HPS EMAC PHY 信号适配到 FPGA 收发器 I/O 管脚处的 SGMII PHY 接口。虽然可以为该适配设计自定义逻辑，但本节主要说明使用 Platform Designer (Standard) 适配器 IP。

指南：使用 Intel HPS GMII to TSE 1000BASE-X/SGMII PCS Bridge, Platform Designer (Standard) 中有该桥接。

在 Platform Designer (Standard) 中配置 EMAC 的 HPS 组件，以作为 “FPGA” I/O 实例。请勿将生成的 HPS 组件 GMII 信号导入 Platform Designer (Standard)。反而需要将 Intel HPS GMII to TSE 1000BASE-X/SGMII PCS Bridge 添加到 Platform Designer (Standard) 子系统并连接 HPS 组件的 GMII 信号。该桥接使用 Platform Designer (Standard) 中的 Intel HPS EMAC Interface Splitter 将 `emac` 管线从 HPS 组件中分离出来，以供 GMII 桥接使用。该桥接例

化以 1000 BASE-X/SGMII PCS PHY-only 模式（例如，无软 MAC 组件）配置的 Intel Triple Speed Ethernet (TSE) MAC。请参阅 [Embedded Peripherals IP User Guide](#) 获得如何使用 Intel HPS GMII to TSE 1000BASE-X/SGMII PCS Bridge 的信息。

注意: 请参阅 [Cyclone V SGMII Example Design](#) 了解本实现的硬件和软件实例。

4.5.1.2.5. MDIO

MDIO PHY 管理总线中每 MAC 有两个信号：MDC 和 MDIO。MDC 是时钟输出，但非自由运行。在 2.5 MHz 时，其最小周期为 400-ns。MDIO 是具有 High-Z 总线转向周期的双向数据信号。

MAC 写入 PHY 时，启动下降沿上的数据，表示有 200 ns - 10 ns = 190 ns 用作接收器的飞行时间，信号设置和建立。因为直到下一个负沿才切换数据，因此还有 200 ns 保持时间。几乎所有电路板拓扑都能轻松满足这些要求。MAC 从 PHY 读取时，PHY 负责将 0 到 300 ns 的读数据输回 MAC，从 100 ns 中减去 10 ns 用于建立，或 90 ns 用作接收器处的飞行时间，信号设置和设置。该要求也很容易满足。

指南：在板上实现 MDC/MDIO 的上拉电阻。

这两种信号都需要外部上拉电阻，通常为 1K，但是 PHY 数据表可能会有所不同。

指南：确保满足接口时序要求。

对于与 MDC 相关的数据，MDIO 有 10ns 的建立和保持时间。

4.5.1.3. 常见 PHY 接口设计考量

4.5.1.3.1. 信号完整性

指南：在 PHY 输出上使用正确板级终端。

很少有 PHY 为其到 Cyclone V/ Arria V SoC 的输出提供 I/O 调整，因此明智的做法是使用模拟器再次仔细检查该信号通路。如有必要，在靠近 PHY 输出管脚的每个信号上放置一个串行电阻器以减少反射。

指南：尽量减少 PHY TX_CLK 和 EMAC RX_CLK 输入上的反射，以防止双倍时钟计时。

由于必须保持信号完整性，所以请注意将该连接布线为“T”字形，以便 REF_CLK 加载时不会看到双沿。请确保将 REF_CLK 加载时的反射减少到最小，以防止双倍时钟计时。

指南：使用 Signal Integrity (SI) 仿真工具。

在单向信号上运行 SI 仿真相当直观。因为这些信号大多数始终是 point-to-point 信号，所以只需要确定一个正确的串行电阻器放置到每个信号上就足够了。多数时候，并不需要这个电阻器，但是在确定电阻器时，应先了解器件驱动强度，走线长度和拓扑结构。

4.5.2. USB 接口设计指南

Cyclone V/ Arria V SoC Hard Processor 系统可使用支持的 1.8V, 2.5V, 3.0V 和 3.3V I/O 标准的 HPS Dedicated I/O 将其嵌入式 USB MAC 直接连接到行业标准 USB 2.0 ULPI PHY。由于不使用 FPGA 路由资源和固定时序，这样就可简化设计。本指南介绍的设计指南涵盖所有支持 PHY 操作速度：High-Speed (HS) 480 Mbps, Full-Speed (FS) 12 Mbps 和 Low-Speed (LS) 1.5 Mbps。

注意: Cyclone V SoC U19 封装 (484 引脚) 仅有一个 USB 控制器可用。

指南: 设计支持两种 **USB PHY** 模式的电路板, 其中由器件提供时钟, 而非外部时钟作为源。

Cyclone V/ Arria V SoC 上 ULPI MAC 和 PHY 之间的接口由 MAC 到 PHY 的 DATA[7:0], DIR 和 NXT, 以及 MAC 到 PHY 的 STP 组成。同时还有一个由 PHY 驱动的 60 MHz 静态时钟, 该时钟是操作时必需使用的时钟, 因为其中包含一些从 HPS 到 USB MAC 的寄存器访问。请确保遵循 PHY 制造商关于 RESET 和上电的建议。

指南: 请确保 **USB** 信号走线长度最小。

在 60 MHz 时, 周期为 16.67 ns, 在此期间, 例如, 时钟必须先从外部 PHY 传输到 MAC, 然后数据和控制信号必须从 MAC 传输到 PHY。因为存在往返传播延迟, 所以 CLK 和 ULPI 信号的最大长度非常重要。基于时序数据, 建议的最长长度应小于 7 英寸 (17.78 厘米)。该要求是基于 5 ns Tco 规格的 PHY。如果使用较慢的规格, 则总长度必须相应缩短。

指南: 请确保考虑信号完整性。

信号完整性非常重要, 尤其在 HPS 子系统下从 PHY 驱动到 MAC 的 CLK 信号中。因为这些信号具有最大长度的点对点信号, 因此通常可以无终点运行, 但建议模拟其走线以确保对其产生的反射最小化。除非存在针对该仿真的其他说明, 否则通常建议使用 FPGA 的 50 欧姆输出设置。如果可能, 可使用 PHY 供应商提供的类似的设置。

指南: 正确设计 **OTG** 操作 (如果 **OTG** 可用)

使用 On-the-Go (OTG) 功能时, SoC 可用作主机或端点。处于主机模式时, 请考虑电源输送, 比如, 支持 USB Flash 驱动, 或者可能支持 USB Hard Drive 时。必须考虑这些电源要求和反向电源, 通常是在使用外部二极管和限流器时, 例如在 Cyclone V SoC 或 Arria V SoC 开发套件中使用。

4.5.3. QSPI Flash 接口设计指南

最多可选择 4 个 QSPI 芯片与 Cyclone V/ Arria V SoC 一起使用。仅可从连接到片选零的 QSPI 对器件进行引导。

指南: 请确保按数字顺序使用 **QSPI_SS** 信号。

Intel Quartus Prime 认为需要按照顺序使用 QSPI_SS 信号。例如, 在没有使用 SS1 的情况下, 就不可能使用 SS0 和 SS2。

注意: 请参阅 [Cyclone V 和 Arria V SoC 支持的闪存器件](#) 获得支持的 QSPI 器件清单。
RocketBoards.org 还在 [GSRD v13.1 - Booting from QSPI](#) 和 [GSRD v13.1 - Programming QSPI Flash](#) 中提供了有用信息。

指南: 如果您的设计使用 **4-byte** 寻址的 **QSPI flash**, 则在设计电路板时请确保 **HPS** 复位的同时 **fQSPI flash** 也是复位状态或重新上电。

Cyclone V 和 Arria V 中默认 HPS 引导 ROM 以 3-byte 寻址模式运行。如果在操作期间将 QSPI flash 切换为 4-byte 寻址, 则请在 HPS 复位的同时确保其已返回默认的 3-byte 寻址模式。否则 HPS 无法从 QSPI 闪存器件进行引导, 或无法访问 QSPI 闪存器件。

可使用以下方法之一将 QSPI 切换回 3-byte 寻址模式：

- 如果 QSPI 器件具有复位管脚，则每次 HPS 器件复位时都要将该复位信号置位。
- 如果 QSPI 器件无复位管脚，则每次 HPS 器件复位时上电 QSPI 器件。RocketBoards.org 的

CV SoC 和 AV Soc QSPI Boot 页面提供实现 QSPI flash 复位信号的建议。请参阅“选择 QSPI Flash 器件”和“对 QSPI 和 SD/MMC/eMMC 提供内存复位”了解更多有关信息。

相关链接

- [选择 QSPI Flash 器件](#) (第 30 页)
- [为 QSPI 和 SD/MMC/eMMC 提供 Flash Memory 复位](#) (第 29 页)
- [Cyclone V SoC 和 Arria V SoC 支持的 Flash 器件](#)

4.5.4. SD/MMC 和 eMMC 卡接口设计指南

指南：如果计划支持 **SD 1.8V** 功能，需要包括一个电压转换器。必需有一个转换器是因为 **HPS I/O** 无法像 **SD** 卡一样动态更改电压电平。

SD 卡的初始工作电压为 3.3V，部分卡能在初始化后切换到 1.8V。此外，一些 MMC 卡在 1.8V 和 3.3V 下都能工作。因为在引导过程中 BSEL 值为常量，所以要求收发器支持对 1.8 V 运行的卡进行电平切换或隔离。

请遵循相应 *硬核处理器系统技术参考手册* “SD/MMC 控制器”章节下“电压切换”中的指导。一些 MMC 卡只可使用 1.8V I/O 操作，并且不需要 3.3V 的初始操作。这样，就不需要电平转换器。

表 8. 电平切换要求

HPS I/O Bank 电压	SD 卡电压	是否需要电平切换器？
3.3V	3.3V	No
3.3V	1.8V	Yes
1.8V	3.3V	Yes
1.8V	1.8V	Yes

指南：请确保考虑初始 **ID** 模式和正常操作的时序。

在处理 ID 过程中，SD 卡最初以最大 400 KHz 运行。此后的数据传输模式中，时钟最高可运行到 12.5 MHz。而正常操作中，时钟的运行频率最高达到 50 MHz。Boot ROM 负责在 ID 和传输模式期间时钟配置正确。

请参阅相应 *硬核处理器系统技术参考手册* 中“引导和配置”附录下的“SD/MMC 控制器 CSEL 设置”表。

指南：请确保每次复位 HPS 时都将 SD/MMC 卡复位。

为了允许系统从 SD/MMC 进行引导，无论 HPS 何时复位，都请确保将 SD/MMC 卡复位。这样就可以保证存储卡处于该引导代码的预期状态。

相关链接

- [电压开关 \(Cyclone V\)](#)
Cyclone V HPS 中 1.8 V SD 操作的水平转换准则

- [电压开关 \(Arria V\)](#)
Arria V HPS 中 1.8 V SD 操作的水平转换准则
- [SD/MMC 控制器的 CSEL 设置](#)
Cyclone V 硬核处理器系统技术参考手册中 “引导和配置” 附录下的表格
- [SD/MMC 控制器的 CSEL 设置](#)
Arria V 硬核处理器系统技术参考手册中 “引导和配置” 附录下的表格

4.5.5. NAND Flash 接口设计指导

指南：请确保选择的 **NAND flash** 器件兼容 **8-bit ONFI 1.0**（或更高版本）器件。

HPS 中的 NAND flash 控制器要求：

- 外部 flash 器件 8-bit ONFI 1.0 兼容
- 单层单元（SLC）或多层单元（MLC）
- 页面大小：512 字节，2 KB，4 KB 或 8 KB
- 每 block 页面大小：32，64，128，256，384 或 512
- 纠错码（ECC）扇区可编程为 512 字节（用于 4-bit，8-bit-或 16-bit 纠正）或 1024 字节（24-bit 纠正）

不可将 NAND 接口导出到 FPGA。

注意：请参阅 [Cyclone V](#) 和 [Arria V SoC](#) 支持的闪存器件来获得支持的 NAND 器件清单。

4.5.6. UART 接口设计指南

指南：通过 **FPGA** 架构路由 **UART** 信号时，正确连接流程控制信号。

通过 FPGA 路由 UART 信号时，流量控制信号可用。如果未使用流量控制，则按照表格中所示连接 FPGA 信号。

表 9. UART 连接以禁用流量控制

信号	方向	连接
CTS	Input	Low
DSR	Input	High
DCD	Input	High
RI	Input	High
DTR	Output	No-connect
RTS	Output	No-connect
OUT1_N	Output	No-connect
OUT2_N	Output	No-connect

4.5.7. I²C 接口设计指南

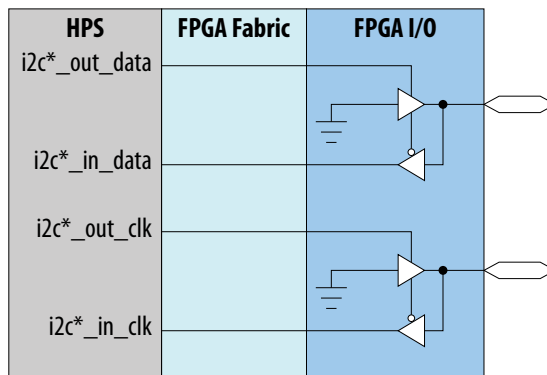
指南：通过 **FPGA** 架构路由 **I²C** 信号时例化开漏缓冲器。

通过 **FPGA** 架构路由 **I²C** 信号时，请注意从 **HPS** 到 **FPGA** 架构的 **I²C** 管脚 (**i2c*_out_data**, **i2c*_out_clk**) 并非开漏且为反向逻辑电平。因此，将逻辑电平 **zero** 驱动到 **I²C** 总线时，这些管脚为有效高电平。该实现非常有用，因为可使用他们直接绑定到三态缓冲器的输出使能。必须使用 **altioobuf** 实现开漏缓冲器。

指南：请确保上拉电阻都已添加到电路板设计中的外部 **SDA** 和 **SCL** 信号中。

因为 **I²C** 信号是开漏信号，因此当总线上没有器件将其拉低时，就需要该信号来确保将总线拉高。

图 6. I²C 有线连接 **FPGA** 管脚



4.5.8. SPI 接口设计指南

指南：考虑将 **SPI** 从接口信号路由到 **FPGA** 架构

由于 Cyclone V/ Arria V SoC 器件中的勘误，造成 **SPI** 输出使能未连接到 **SPI** **HPS** 管脚。由此，**HPS** **SPIS_TXD** 管脚无法在 **ctrlr0** 寄存器中的 **slv_oe** 位 (**bit 10**) 设置为 1 时成为三态。

将 **SPI** **Slave** 信号路由到 **FPGA** 会显露输出使能信号，并允许将其连接到 **FPGA** 三态管脚。

指南：如果您的 **SPI** 外设需要 **SPI** 主从选择在整个事务期间保持低电平，可考虑使用 **GPIO** 作为从选择，或配置 **SPI** 主选择以在事务期间置位从选择。

默认情况下，将 **SPI** 主选择配置为 **ctrlr0.scph = 0** 和 **ctrlr0.scpol = 0**，使得 Cyclone V 或 Arria V **HPS** **SPI** 主选择解除置位每个数据字之间的从选择信号。将 **ctrlr0.scph** 设置到 1，**ctrlr0.scpol** 也设置到 1 时，整个传输期间 **SPI** 主选择置位从选择。

或者，考虑将 **SPI** 主外设路由到 **FPGA**，并使用 **GPIO** 控制从选择信号。

注意：如果使用该方法，则请参阅 Knowledge Base 中的文章，如下：

- 为什么通过 **FPGA** 逻辑将 **HPS** 外设时钟连接到外部管脚会导致 **Quartus** 适配器错误？
- 为什么将从选择映射到 **GPIO** 后 **HPS** **SPI** **Master** 会失败？

相关链接

- [ctrlr0](#)
有关 Cyclone V HPS 中 `ctrlr0.scph` 和 `ctrlr0.scpol` 位的详细信息，请参阅 *Cyclone V 硬核处理器技术参考手册* “SPI Master” 章节中的 “ctrlr0” 部分。
- [ctrlr0](#)
有关 Arria V HPS 中 `ctrlr0.scph` 和 `ctrlr0.scpol` 位的详细信息，请参阅 *Arria V 硬核处理器技术参考手册* “SPI Master” 章节中的 “ctrlr0” 部分。

5. SoC FPGA 的嵌入式软件设计指南

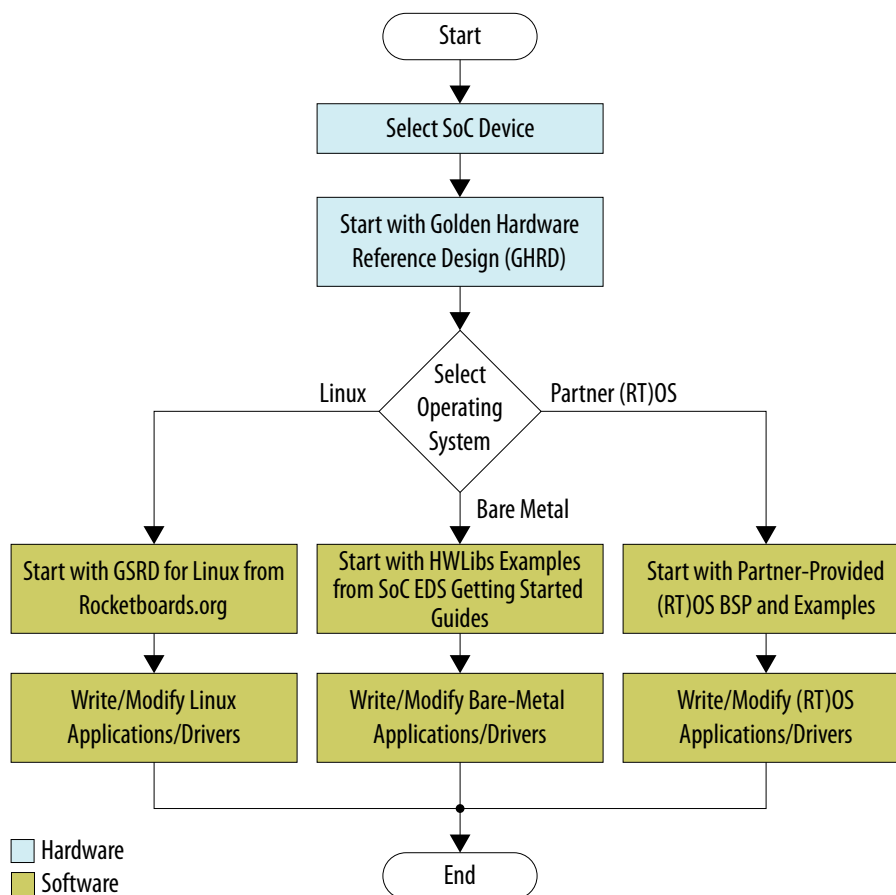
5.1. HPS 的嵌入式软件：设计指南

5.1.1. 组装软件开发平台的组件

为了成功构建您的软件开发平台，Intel 建议从基线工程开始，因为它是 HPS 系统的已知良好配置。然后修改基线工程以适合您的应用程序。

下图显示的建议程序，用来确定软件开发平台组件。

图 7. 组装软件开发平台



总之，该处理过程包括以下步骤：

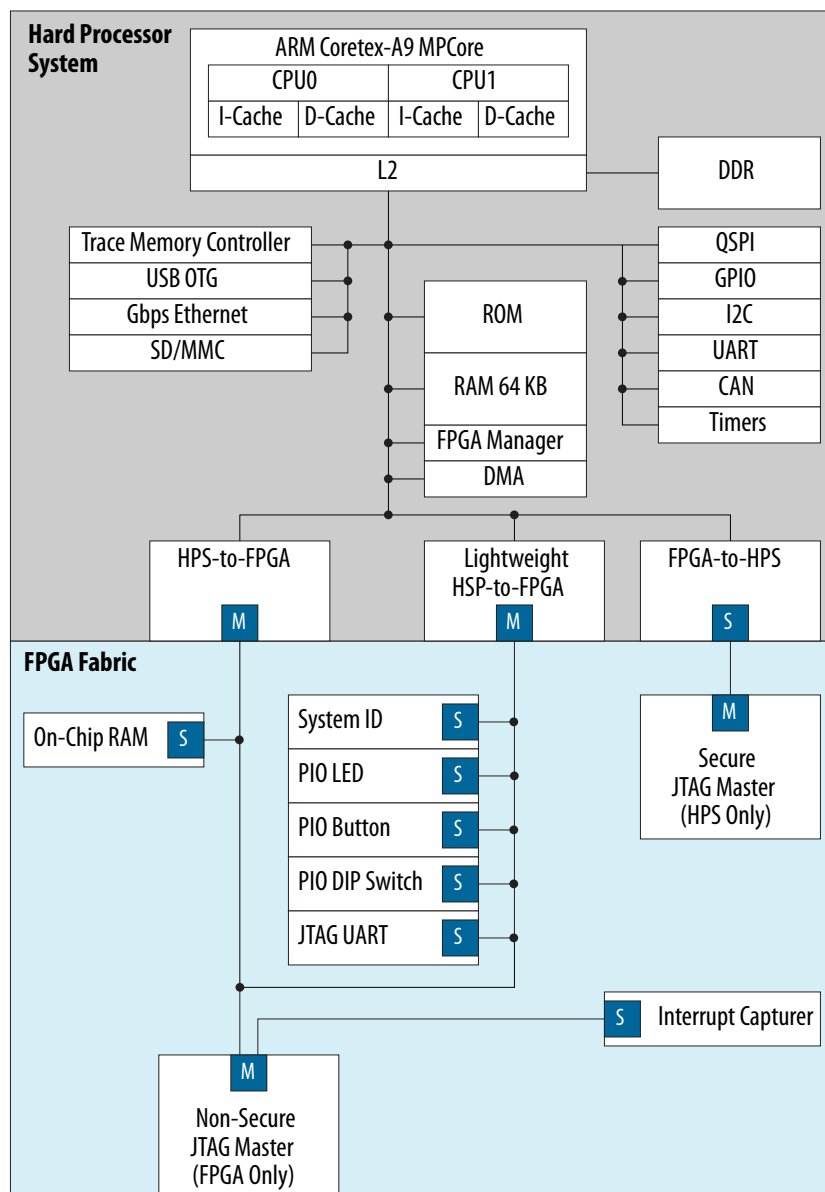
1. 选择所需的器件
2. 使用 GHRD 作为硬件工程起点
3. 选择操作系统：裸机，Linux*或合作伙伴实时操作系统（RTOS）
4. 写入或者更新终端应用程序以及驱动程序

5.1.1.1. Golden Hardware Reference Design

Golden Hardware Reference Design 是一个 Intel Quartus Prime 工程，其中包含 Cyclone V SoC / Arria V SoC Development Kit 的完整 HPS 设计。GHRD 中有与引导源，SDRAM 存储器和开发板上其他外设的连接。

每个新版本的 SoC EDS 中，其 SoC EDS 工具都包含 GHRD。已根据 Intel Quartus Prime Design Software 的每个重大发布对 GHRD 进行了回归测试，并包括已知硬件问题的最新错误修复。GHRD 用作 SoC FPGA 硬件系统的已知良好配置。

图 8. Cyclone V / Arria V SoC Golden Hardware Reference Design 概述



在 FPGA 架构中 GHRD 具有最少外设集，因为 HPS 提供了大量外设选择。HPS-to-FPGA 和 FPGA-to-HPS 接口被配置为 64-bit 数据宽。

指南：Intel 建议使用最新 GHRD 作为新的 SoC FPGA 硬件工程的基线。然后修改该设计以适合您的应用程序端。

GHRD 可从以下位置获得：

- 最新版 [GSRD for Linux page](#)，是已知的最佳配置。
- <SoC EDS installation directory> \examples\hardware\cv_soc_devkit_ghrd - 相应 SoC EDS 版本支持的版本，用作所提供 HWLibs 设计实例的基础。

5.1.2. 为应用程序选择操作系统

5.1.2.1. Linux 或 RTOS

有许多操作系统都支持 Cyclone V SoC 和 Arria V SoC，包括 Linux 和多个实时操作系统 (RTOS)。有关 Intel 的 SoC Partner OS 生态系统的更多信息，请访问 [Intel FPGAs 网页的 Ecosystem](#) 选项卡。

Partner OS 提供商为 SoC FPGA 器件提供电路板支持包和商业支持。Linux 社群还为 SoC FPGA 器件提供板级支持包和开发社群支持。

影响选择 SoC FPGA 操作系统的因素有很多，包括操作系统的特性，许可条款，基于 OS 的可协作软件工程和框架，可用的器件驱动器和参考软件，内部旧版代码和对 OS 的熟悉度，系统的实时要求，功能安全性和应用程序要求的其他认证。

为您的应用程序选择合适的 OS，建议您熟悉可用于 SoC FPGA 的商业和开源操作系统所提供的功能和提供的服务。Intel 的 OS 合作伙伴，行业网站是一个很好的信息来源，有助于您做出选择。

谈及操作系统与裸机应用程序的实时性能，存在许多误解。对于 Cortex A 级处理器，除了提供管理运行时间应用程序的工具外，实时操作系统还提供了许多功能，可有效利用处理器资源。您可能会发现这些效率的产生为应用程序带来足够高的实时性能，从而使您能够继承大量可用的器件驱动程序，中间件数据包，软件应用程序和支持的服务。所以在选择操作系统时考虑这一点非常重要。

5.1.2.2. Bare Metal (裸机)

HPS 可用于裸机配置 (无需 OS)，同时 Intel 提供的 HWLibs (Hardware Libraries)，其中包含用于大多数 HPS 外设的高级 API，和低级宏。

但是，要将裸机应用程序用于 HPS，您必须熟悉开发时间运行能力，以确保您的裸机应用程序有效使用 MPU 子系统内的可用资源。

例如：

- 典型的裸机应用程序仅使用单个内核，如果想充分利用 MPU 子系统，则必须开发运行时间能力来管理内核与 cache 子系统之间的处理进程。
- 随着应用程序复杂性的不断增加，可能需要构建管理和调度进程、处理进程间通信以及应用程序内各事件之间同步的能力。

即使是小型轻量级 RTOS 也能提供简单的调度，进程间通信和中断处理功能，从而有效利用 MPU 子系统内的资源。

5.1.2.3. 使用 Symmetrical 与 Asymmetrical Multiprocessing (SMP vs. AMP) 模式

Cyclone V / Arria V HPS 中的 Dual Core ARM Cortex-A9 MPCore* 可以支持 Symmetrical Multi-processing (SMP) 和 Asymmetrical Multi-processing (AMP) 配置模式。

SMP 模式中，单个 OS 实例控制两个内核。SMP 配置被各系统制造商的广泛支持，也是多处理的最常见和最直接的配置。

商业开发的操作系统提供的功能可以充分利用 CPU 内核资源，并以有效方式使用这些资源，从而实现最佳性能和易用性。例如，已使用 SMP 的操作系统会提供设置处理器关联的选项。就意味着可将每个任务/线程分配到特定内核上运行。该功能支持软件开发人员更好地控制对每个 Cortex-A9 内核的负载分配，并使系统在替代 AMP 运行时能更快响应。

指南：熟悉商业操作系统中的性能和优化，了解 SMP 使能的 OS 或 RTOS 是否满足您的性能和实时要求。

AMP (Asymmetrical Multi-Processing) 配置中，两个不同的操作系统或单个操作系统的两个实例运行在两个内核上。这两个操作系统本身并不了解他们如何共享 CPU 资源。为确保该环境下有效使用 MPU 子系统资源，那么在设计系统时必须处理几个复杂的问题。

警告：使用 AMP 之前，必须熟悉的技术包括管理和调度进程，处理进程间通信，事件之间同步，管理两个操作系统实例之间安全进程。

注意：OS 提供商通常不支持在 AMP 模式下使用他们的操作系统，因此在这种情况下通常需要特殊的支持协议。

5.1.3. 组装用于 Linux 的 Software Development Platform

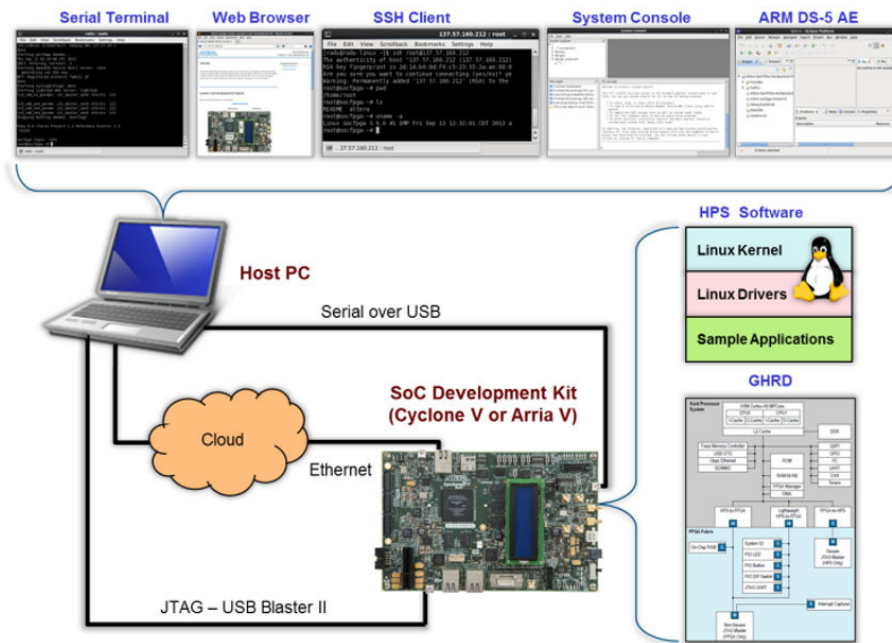
本小节介绍在选择 Linux 作为您端应用程序的 OS 时需要使用的设计指南。

5.1.3.1. 面向 Linux 的黄金系统参考设计 (GSRD)

在针对 Linux 提供的 Golden System Reference Design (GSRD) 中，包含以下内容：

- GHRD (Golden Hardware 参考设计) - Quartus Prime 工程
- 基于 U-Boot 的 Bootloader 参考
- Linux BSP 参考
- Linux 应用程序示例

图 9. 面向 Linux 的 GSRD - 概述



GSRD for Linux 是一个经过充分测试的设计，也是知名的优秀设计，展示了同时使用 HPS 和 FPGA 资源的系统，如何用作基线工程。

指南：要成功构建您的软件开发平台，建议使用 **GSRD** 作为基线工程，然后对其进行修改以符合您的应用程序需求。

GSRD 的对象是 Intel SoC Development Boards，并以源代码和预编译两种形式提供，并可从 [GSRD User Manuals](#) 得到。

指南：建议所有新工程都使用最新版本的 **GSRD** 作为基线。

5.1.3.2. 源代码管理考量

GSRD 构建过程依赖于线上可用的几个 git 树，包括：

表 10. Git Tree 链接

Git Tree	链接
Intel SoC FPGA Linux Kernel	https://github.com/altera-opensource/linux-socfpga
Intel SoC FPGA Linux designs	https://github.com/altera-opensource/linux-refdesigns
Intel SoC FPGA Angstrom recipes	https://github.com/altera-opensource/angstrom-socfpga

注意：Intel 提供 Linux 启动，上游到主流，以及与 Linux 社群合作。Intel 提供两个内核版本，最新的 stable kernel (N)和最新的 LTSI kernel (M) 同时停止支持早前的 Linux 内核版本 (N-1, M-1)。但还是可以在任何时候从内核存储库中获得 (N, N-1, M, M-1) 版本。已删除较早以前的内核版本。

指南：管理您的 **Git** 存储库，并且请勿认为 **altera** 开源网站上的存储库内容任然可用。可通过多种方式管理 **Git** 存储库，例如使用 **Git** 服务供应商。管理您的 **Git** 存储库的益处包括，构建可再生性，源代码管理以及利用 **Git** 支持的分布式模型。

GSRD 使用 **Angstrom rootfilesystem**，并通过 **Yocto recipes** 构建。**Recipes** 引入各种开源数据源，并将它们构建成 **rootfilesystem**。由于其中一些 **recipes** 是通用的，并且不涉及特定版本，因此每个构建的最终结果可能会各不相同。

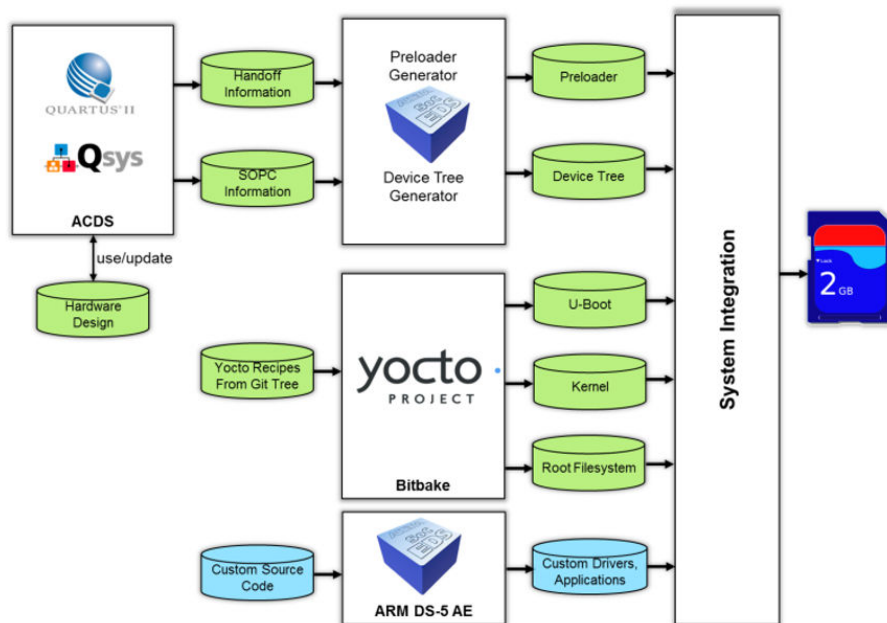
指南：如果重新构建 **Yocto rootfilesystem** 后并需要保持重复性，就必须将用于构建的 **Yocto** 下载文件夹保留一个副本。

指南：如果重新构建 **Angstrom rootfilesystem** 后还需要保持重复性，就必须将用于构建的 **Yocto** 下载文件夹保留一个副本。

5.1.3.3. GSRD for Linux 开发流程

下图显示为基于 GSRD 工程开发流程的高级视图。请从下方的 **GSRD User Manuals** 链接获得详细信息。

图 10. GSRD for Linux - 开发流程



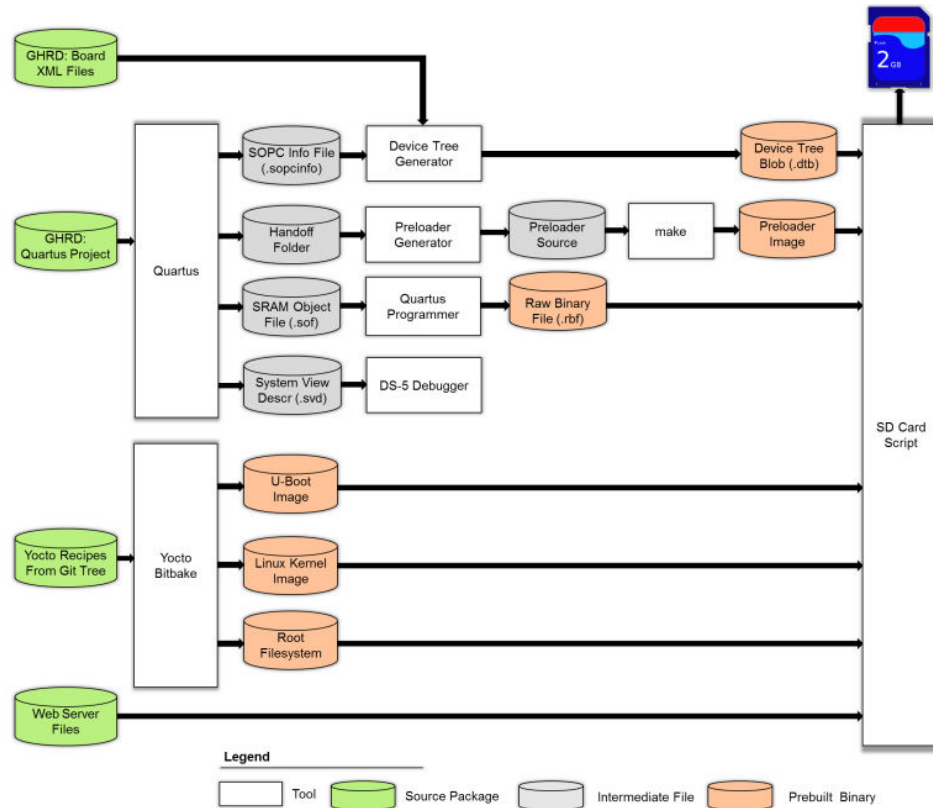
相关链接

[GSRD 用户手册](#)

5.1.3.4. GSRD for Linux 创建流程

下图为 GSRD 的详细构建流程图。请从下方的 **GSRD User Manuals** 链接获得更多详细信息。

图 11. GSRD for Linux - 创建流程



上述构建流程虽然用于 GSRD for Linux，但也可按照每个工程的个体需求进行调整。例如：

- 不使用 Yocto Bitbake 的情况下单独构建 Linux 核心。
- 不使用 Yocto Bitbake 的情况下单独构建 Linux 文件系统。
- 不使用 Device Tree Generator 的情况下管理 Linux Device Tree。例如，可以手动编辑。

相关链接

[GSRD 用户手册](#)

5.1.3.5. Linux Device Tree 设计考量

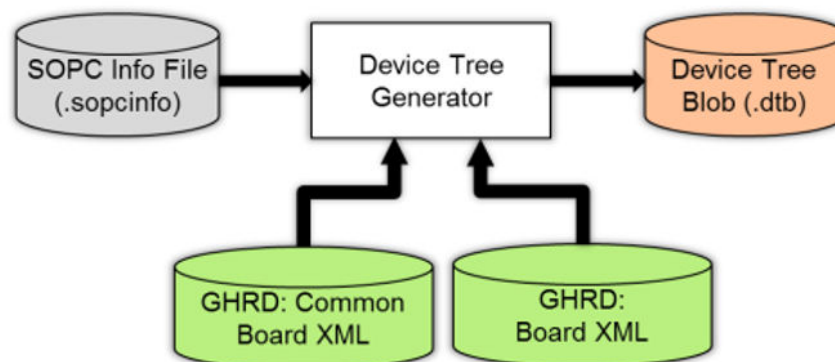
Linux Device Tree 是一种数据结构，向 Linux 操作系统核心说明底层硬件。通过将该数据结构传递到 OS 核心，单个 OS 二进制可以支持硬件的多种类型。当硬件中包含 FPGA 时，这种灵活性就显得格外重要。

管理 Linux Device Tree 的建议程序是：

1. 从 SoC FPGA 参考 Device Trees 开始，该部分在 Intel SoC 开发套件的 Linux 核心源代码中提供。他们涵盖了器件的 HPS 部分，但未覆盖随工程而变化的 FPGA 部分。SD/MMC 和 QSPI 版本和核心源代码一起提供。
2. 对比 Intel SoC 开发套件，根据需要编辑 Device Tree 以接纳电路板的变化。
3. 根据需要编辑 Device Tree，以适应面向 FPGA Soft IP 的 Linux 驱动器。

注意: GSRD for Linux 使用一种不同的流程, 上述建议的流程依赖于一个名为“Linux Device Tree Generator”的自定义工具, 该工具是作为 SoC EDS 的部分一起提供。

图 12. GSRD for Linux 的 Device Tree Generation 流程



请从以下提供的 **DeviceTree Generator User Guide** 链接了解有关 Linux Device Tree Generator 的更多详细信息。

相关链接

[DeviceTree Generator 用户指南](#)

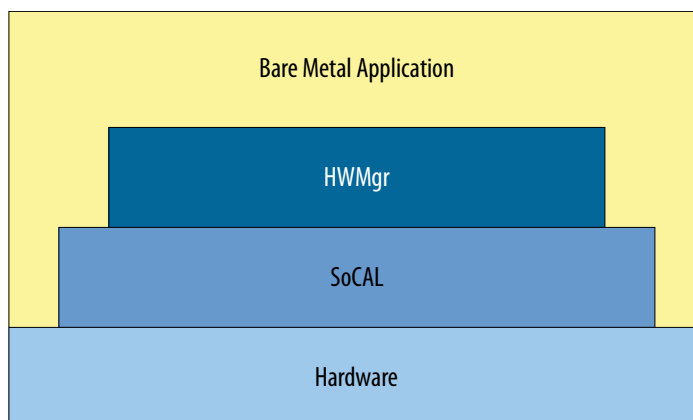
5.1.4. 组装 Bare-Metal 应用程序的 Software Development Platform

Intel 硬件库 (HWLibs) 是随 SoC EDS 和 HPS 的各种组件一同提供的低级裸机软件库。Intel 的 OS 合作伙伴同场也会使用 HWLibs 构建操作系统的板级支持包。

HWLibs 有两个组件:

- SoC Abstraction Layer (SoCAL): 符号寄存器抽象随后使能对地址空间内 HPS 器件寄存器的直接访问和控制。
- Hardware Manager (HWMgr): API 提供更复杂的功能和更高级别用例情景的驱动程序。

图 13. HWLibs 概述



需注意不是所有硬件都包含在 SoCAL 和 HWMgr 中，因此需要根据应用程编写自定义代码。使用 HWLibs 的软件应用程序应该有运行时间规定以管理 MPU 子系统资源、cache 和存储器。这些规定通常是操作系统提供。

指南：建议仅当您熟悉管理应用程序的开发运行时间规定时，才使用 HWLibs。

指南：使用 HWLibs Project Generator 创建您的自定义 HWLibs 工程。

Intel 建议使用 RocketBoards 上提供的 HWLibs Project Generator 工具创建您的自定义 HWLibs 工程。

相关链接

- [Intel® SoC FPGA 嵌入式开发套件用户指南](#)
- [Cyclone V Bare Metal Example Using SoC EDS Standard Edition](#)
- [HwLibs 裸机开发入门](#)
- [设计实例](#)
其他 HWLibs 实例

5.1.5. 组装用于合作伙伴 OS 或 RTOS 的软件开发平台

合作伙伴 OS 提供商为 SoC FPGA 器件提供板级支持保和商业支持。通常，合作伙伴支持包括实例入门工程和相关文档。

注意：请参阅合作伙伴文档和支持服务了解针对合作伙伴 OS 或 RTOS 时如何组装软件开发平台。

5.1.6. 选择引导加载程序软件

Cyclone V / Arria V SoC 引导流程包括如下阶段：

1. Boot ROM
2. Preloader
3. Bootloader
4. 实时操作系统或裸机应用程序

图 14. Cyclone V / Arria V SoC 引导流程



所有 Cyclone V SoC / Arria V SoC 应用程序都需要 BootROM 和 Preloader 阶段。GSRD 使用 U-boot 和 Linux，但自定义应用程序可能会实现不同的流程，例如使用 Preloader 直接加载裸机应用程序。

通常，Preloader 的主要职责是：

- 执行其他 HPS 初始化
- 启动 SDRAM
- 将下一个引导阶段从 Flash 加载到 SDRAM 并跳转到该阶段

目前，有两个不同的 Preloader 选项可用：

- SPL - U-Boot 的一部分。随 SoC EDS 提供，使用 GPL (Open Source) License
- MPL - 随 SoC EDS 提供，作为使用 HWLibs (bare-metal libraries) 的实例。使用 BSD 许可。

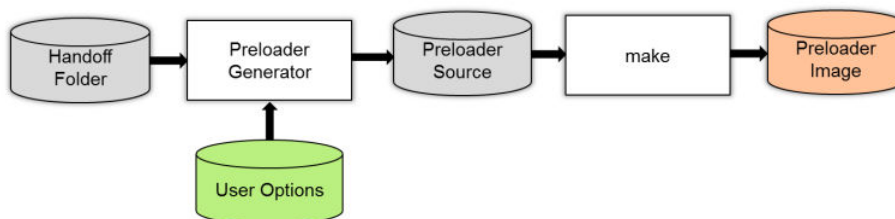
注意： Preloader 需要一个特殊的头 (header) 放置于下一阶段引导影响的开头。此外，该 header 包含一个用于验证映像的 CRC 值。可使用 SoC EDS 中的 mkimage 实用程序将该 header 附加到映像。

Bootloader 的典型职责与 Preloader 相似，只是它不需要启动 SDRAM。因为 Bootloader 已经置于 SDRAM 中，所以不受 OCRAM 大小的限制。因此，它可以提供很多功能，例如网络堆栈支持。

典型 HPS 系统中有许多寄存器，需要根据 MPU 子系统内的给定配置，片上网络互连组件，SDRAM 存储器，flash 引导源和外设接口对他们进行设置。关于引导或初始化目的设置封装在如下位置：

- RBF File(s) - 包含 SDRAM 的寄存器设置以及专用 I/O 和 FPGA 管脚配置。
- U-Boot 源代码 - 用于其余设置

图 15. Preloader 构建流程



注意： 强烈建议使用 bsp-editor 生成 Preloader。还建议（但不是必需）从相同源代码构建 U-Boot。

5.1.7. 选择用于开发，调试和跟踪的软件工具。

注意： 使用特定 Partner OS 或 RTOS 时，请咨询 OS 经销商并参阅 OS 文档了解是否需要任何特定的工具。有些 OS 经销商甚至提供建议与 OS 一起使用的全套工具。

注意： 熟悉可用的开发，编译和调试工具。Intel FPGAs 网页的 Ecosystem 选项卡下提供所支持工具的列表。

5.1.7.1. 选择软件构建工具

指南： 决定要使用的软件开发工具，并选择工具的版本。

软件开发工具包括编译器编译器，编程文件生成工具，链接器和归档程序。Arm Development Studio 5* (DS-5*) Intel SoC FPGA Edition 包含如下软件创建工具：

- ARMCC Bare-metal Compiler
- Mentor Graphics CodeSourcery Lite GCC-based bare-metal Compiler
- Linux Linaro Compiler

还有来自第三方提供者的其他开发工具。

5.1.7.2. 选择软件调试工具

指南：选择软件调试工具。

Arm DS-5 Intel SoC FPGA Edition 包括一个功能齐全并基于 Eclipse 的调试环境。还有来自第三方提供的其他调试工具，例如 Lauterbach T32。

该调试工具需要一个 SoC FPGA 器件的 JTAG 连接。可通过以下几种方式实现该连接：

- 可使用板上可用的嵌入式 USB-Blaster II 芯片，例如 Cyclone V SoC / Arria V SoC Development Kit。
- 使用 Lauterbach T32 工具时可能需要外部 JTAG 硬件。

5.1.7.3. 选择软件跟踪工具

追踪功能非常有助于分析性能瓶颈，调试崩溃情况和调试复杂案例。可以两种方式进行跟踪：

- **Non-real-time**：通过将跟踪数据存储在系统存储器中（例如 SDRAM）或嵌入式跟踪缓冲区，然后停止系统运行，下载跟踪信息并对其进行分析。
- **Real-time**：使用外部适配器跟踪“跟踪端口”处的数据。需要目标板支持这种使用方式。

通常，调试工具还能跟踪嵌入式软件程序执行，但可能需要外部硬件。例如，随 SoC EDS 提供的 DS-5 支持“non-real-time”（非实时和“real-time”（实时）跟踪。用于实时跟踪时，需要名为“DSTREAM”的外部跟踪工具。Lauterbach T32 与其类似，也需要使用外部硬件进行实时跟踪。

5.2. Flash 器件驱动设计考量

SoC FPGA 支持以下 flash 器件类型：QSPI，NAND，SD/MMC/eMMC。

注意：

请参阅 [Supported Flash Devices for Cyclone V 和 Arria V SoC](#) 了解支持的 flash 器件列表。使用“Intel Tested and Supported”器件，最大限度减少开发工作和不兼容的风险。另一个最佳选项是选择“Known to Work”（已知可工作）器件。意味着兼容 BootROM 的器件，并且已证明可与至少一个 Bootloader 一起工作，但它可能并不是您需要的 Bootloader。它还可能没有 HWLibs，OS Support 或 HPS Flash Programmer Support。

5.3. SD 卡低功耗模式设计考量

SD/MMC Controller 具有低功耗模式，将寄存器 **clkena** 的 **cclk_low_power** 位设置为 1 使能该模式。该模式生效后，如果 SD 卡空闲至少 8 个卡时钟周期，则该卡的时钟被禁用。

该低功耗模式期间，SD I/O 信号的状态如下：

- SD_CLK = 0
- SD_CMD = 1
- SD_D0..3 = 悬空

如果端应用程序要求低功耗模式下所有 SD I/O 信号悬空，则建议执行以下步骤：

1. 未使用该卡时，在最后一个命令之后：

- 使用 **gpio** 寄存器将与 SD_CMD and SD_CLK 管脚相关的 GPIO 设置为输入。
 - 使用 **sysmgr.pinmux registers2** 将 SD_CLK 和 SD_CMD 管脚的管脚复用更改为 GPIO 信号。
2. 再次使用该卡时，在下一个命令之前：
- 使用 **sysmgr.pinmux** 寄存器将 SD_CLK 和 SD_CMD 信号的管脚复用改回至 SD I/O 信号。

注意: SD/MMC controller 处于复位状态时，SD I/O 信号的状态如下：

- SD_CLK = 0
- SD_CMD = 悬空
- SD_D0...3 = 悬空

5.4. HPS ECC 设计考量

在所有 RAM 的整个 HPS 子系统中都会使用 ECC，其中包括，外部 HPS EMIF，L2 cache 数据 RAM 和所有外设 RAM。控制器 ECC 采用标准 Hamming 逻辑检测和纠正单 bit 错误，以及检测双 bit 错误。为 Cortex-A9 MPCore L1 cache 存储器和 L2 标记 RAM 提供奇偶校验保护。可选择使能 HPS EMIF 和内部 HPS RAM 上的 ECC。软件控制中有诊断测试模式和错误注入功能。上电或冷复位后默认禁用 ECC。

按照 BSP 生成过程中用户的选项，使用生成的引导码来配置，初始化和使能 ECC。Intel 提供的 HWLibs 库中有访问 ECC 功能的自定义固件和裸机应用程序编码，为编程 HPS 功能提供简单的 API。

更多有关信息，请参阅 [Intel SoC FPGA 嵌入式开发套件用户指南](#) 中的“引导工具用户指南”和“硬件库”章节。

相关链接

[Intel® SoC FPGA 嵌入式开发套件用户指南](#)

5.4.1. 常规 ECC 设计考量

HPS 子系统中的 Each RAM 有自己的 ECC 控制器，以及一组独特功能和要求。但是，仍然需要考虑一些常规的系统集成问题。

5.4.2. 系统级 ECC 控制，状态和中断管理

System Manager 包含一组 ECC 相关寄存器，用于管理 HPS 子系统中所有 ECC 控制器的系统级控制和状态。ECC 相关的中断也是通过这组寄存器进行管理。

相关链接

- [System Manager - Cyclone V Hard Processor System Technical Reference Manual](#)
- [System Manager - Arria V Hard Processor System Technical Reference Manual](#)

5.4.3. L2 高速缓存数据存储器的 ECC

L2 cache memory 由 ECC 保护，而标记 RAM 被奇偶校验保护。L2 cache ECC 是通过 System Manager 中的控制寄存器使能。

有关 L2 cache ECC controller 的详细信息，请参阅相应 *Hard Processor System Technical Reference Manual* 中 "Cortex-A9 Microprocessor Unit Subsystem" 章节的以下部分：

- “Single Event Upset Protection”
- “L2 Cache Controller Address Map for Cyclone V” 或 “L2 Cache Controller Address Map for Arria V”

指南：对于已使能 ECC 的任何可高速缓存的存储区域，L1 和 L2 cache 必须配置为写回 (write-back) 和写分配 (write-allocate)。

对于 Intel SoC FPGA EDS 支持的 BSP，可使用 bsp-editor 实用工具配置 BSP 以获得 ECC 支持。

关于裸机固件，请参阅相应 *Hard Processor System Technical Reference Manual* 中 "Cortex-A9 Microprocessor Unit Subsystem" 章节的 “L2 Cache Controller Address Map” 部分。

指南：使能 L2 Cache Controller 中的 ECC 后，使用 ACP 并通过 L3 互连的高速缓存一致性访问必须运行 64-bit 宽，64-bit 对齐的写访问。

使能 ECC 不会影响 L2 cache 的性能，但是使用 ACP 的访问在存储器中必须为 64-bit 宽，64-bit 对齐。这其中包括通过 FPGA-to-HPS Bridge 访问 ACP 的 FPGA 主控。有一个表格罗列了有关桥接宽度和 FPGA 主控宽度、对齐和突发大小与长度的可能组合，请参阅相应 *Hard Processor System Technical Reference Manual* 中 “HPS-FPGA Bridges” 章节的 “FPGA-to-HPS Access to ACP” 部分。

相关链接

- [Cyclone V 硬核处理器系统技术参考手册](#)
- [Arria V 硬核处理器系统技术参考手册](#)

5.4.4. Flash Memory 的 ECC

HPS 子系统中的所有外设 RAM 都是 ECC 保护。从 flash 器件页面缓冲区运行 read-modify-write (读取-修改-写入) 时，不使用 NAND flash controller ECC 硬件。在 read-modify-write 操作过程中，软件必须更新 ECC。在与硬件 ECC 一起工作的 read-modify-write 操作中，一定是将整个页面读入系统存储器中，并且经过修改后再写回 flash，不依赖 flash 器件的 read-modify-write 功能。

NAND flash controller 无法在运行 copy-back 命令过程中进行 ECC 验证。该 flash controller 复制 ECC 数据，但是在复制过程中并不会对其进行验证。

5.5. HPS SDRAM 考量

5.5.1. 使用 Preloader 调试 HPS SDRAM

要调试 HPS EMIF，可更改 preloader 中的设置 使能运行时间校准报告，调试级别信息并检查 HPS SDRAM PLL 状态。

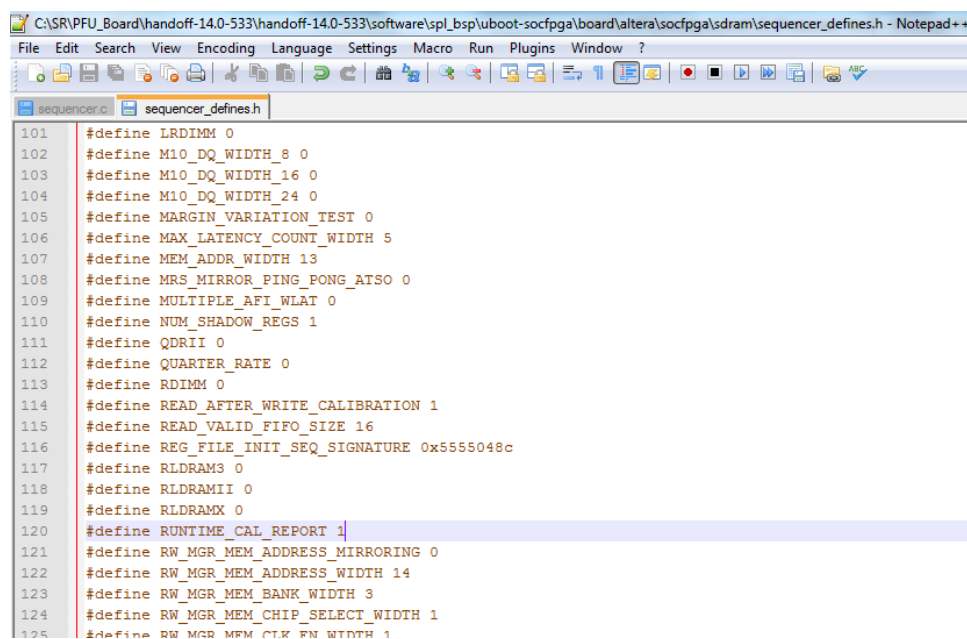
注意: 请参阅 [Intel SoC FPGA Embedded Development Suite User Guide](#) 中的 “Building the Second Stage Bootloader” 了解编译与加载器的分步说明。

相关链接

[Intel® SoC FPGA 嵌入式开发套件用户指南](#)

5.5.1.1. 使能 Runtime Calibration Report

要使能运行时间校验报告, 请使用您的首选编辑器打开 <project_folder>\software\spl_bsp\uboot-socfpga\board\altera\socfpga\sdr\sequencer_defines.h 文件并将 RUNTIME_CAL_REPORT 值配置为 1。



```
101 #define LRDIMM 0
102 #define M10_DQ_WIDTH_8 0
103 #define M10_DQ_WIDTH_16 0
104 #define M10_DQ_WIDTH_24 0
105 #define MARGIN_VARIATION_TEST 0
106 #define MAX_LATENCY_COUNT_WIDTH 5
107 #define MEM_ADDR_WIDTH 13
108 #define MRS_MIRROR_PING_PONG_ATSO 0
109 #define MULTIPLE_AFI_WLAT 0
110 #define NUM_SHADOW_REGS 1
111 #define QDRII 0
112 #define QUARTER_RATE 0
113 #define RDIMM 0
114 #define READ_AFTER_WRITE_CALIBRATION 1
115 #define READ_VALID_FIFO_SIZE 16
116 #define REG_FILE_INIT_SEQ_SIGNATURE 0x5555048c
117 #define RLD3 0
118 #define RLD2 0
119 #define RLD1 0
120 #define RUNTIME_CAL_REPORT 1
121 #define RW_MGR_MEM_ADDRESS_MIRRORING 0
122 #define RW_MGR_MEM_ADDRESS_WIDTH 14
123 #define RW_MGR_MEM_BANK_WIDTH 3
124 #define RW_MGR_MEM_CHIP_SELECT_WIDTH 1
125 #define RW_MGR_MEM_CLK_FN_WIDTH 1
```

5.5.1.2. 更改 DLEVEL 获得更多调试信息

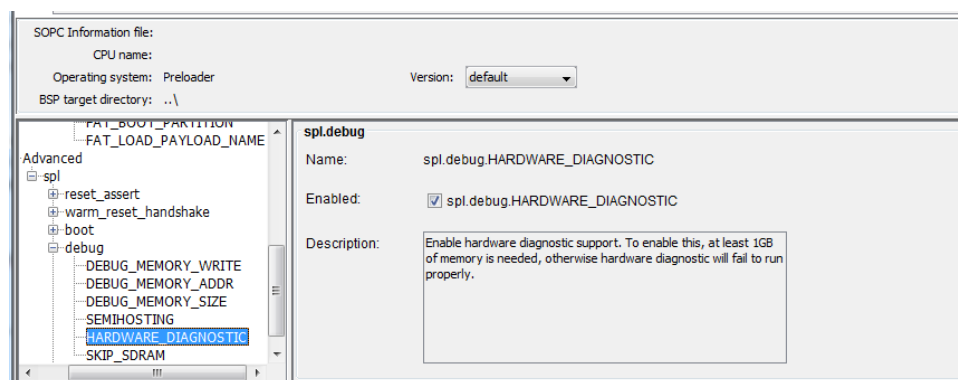
```

C:\SRV\FPU_Board\handoff-14.0-533\handoff-14.0-533\hps_isw_handoff\vedpepper_sys_hps_0\sequencer.c - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
sequencer.c sequencer_defines.h
280      snprintf((char*) (debug_printf_output->active_word), PRINTF_READ_BUFFER_SIZE*4, "INFO:" fmt, ## args);
281      wait_printf_queue();
282
283      #define BFM_GBL_SET(field,value)      bfm_gbl.field = value
284      #define BFM_GBL_GET(field)           bfm_gbl.field
285      #define BFM_STAGE_SET(label)         BFM_GBL_SET(stage,label)
286      #define BFM_INC_VFIFO                bfm_gbl.vfifo_idx = (bfm_gbl.vfifo_idx + 1) % VFIFO_SIZE
287      #define COV(label)
288
289      #elif HPS_HW //
290      // For HPS running on actual hardware
291
292
293      #define DLEVEL 2
294      #ifndef HPS_HW_SERIAL_SUPPORT
295      // space around comma is required for varargs macro to remove comma if args is empty
296      #define DPRINT(level, fmt, args...)    if (DLEVEL >= (level)) printf("SEQ.C: " fmt "\n", ## args)
297      #define IPRINT(fmt, args...)           printf("SEQ.C: " fmt "\n", ## args)
298      #if RUNTIME_CAL_REPORT
299      #define RPRINT(fmt, args...)            printf("SEQ.C: " fmt "\n", ## args)
300      #endif
301      #else
302      #define DPRINT(level, fmt, args...)
303      #define IPRINT(fmt, args...)
304      #endif

```

5.5.1.3. 使能 HPS SDRAM 的 Example 驱动

- 通过 bsp-editor 中的 Hardware Diagnostic Option 使能。请注意：仅 Intel Quartus Prime 14.0 或更高版本中具有 Example driver。
 - PRBS31 Data 码型
 - 写入随机地址=>从随机地址读取
 - 可更改 spl.c 中的参数，选择不同的覆盖范围。



```
672     sdram_size = sdram_calculate_size();
673 #else
674     sdram_size = PHYS_SDRAM_1_SIZE;
675 #endif
676     printf("SDRAM: %ld MiB\n", (sdram_size >> 20));
677
678 #if (CONFIG_PRELOADER_HARDWARE_DIAGNOSTIC == 1)
679     /* Sanity check ensure correct SDRAM size specified */
680     puts("SDRAM: Ensuring specified SDRAM size is correct ...");
681     if (get_ram_size(0, sdram_size) != sdram_size) {
682         puts("failed\n");
683         hang();
684     }
685     puts("passed\n");
686     /*
687      * A simple sdram memory test
688      * If you want more coverage, change the argument as below
689      * SDRAM_TEST_FAST -> quick test which run around 5s
690      * SDRAM_TEST_NORMAL -> normal test which run around 30s
691      * SDRAM_TEST_LONG -> long test which run in minutes
692      */
693     if (hps_emif_diag_test(SDRAM_TEST_FAST, 0, sdram_size) == 0)
694         hang();
695 #endif /* CONFIG_PRELOADER_HARDWARE_DIAGNOSTIC */
696
```

5.5.1.4. 更改 Example Driver 中的 Data 码型

1. sdram_test.c 的路径: <project_folder>\software\spl_bsp\uboot-socfpga\arch\arm\cpu\armv7\socfpga\sdram_test.c
2. 更改 test_rand_address 功能

```
137 /*
138  * Valid arguments for coverage
139  * SDRAM_TEST_FAST -> quick test which run around 5s
140  * SDRAM_TEST_NORMAL -> normal test which run around 30s
141  * SDRAM_TEST_LONG -> long test which run in minutes
142  */
143 int hps_emif_diag_test(int coverage, unsigned int addr_bgn,
144     unsigned int addr_end)
145 {
146     int cnt_max = 1000000, status;
147     unsigned int seed = 0xdeadbeef;
148
149     if (coverage == SDRAM_TEST_NORMAL)
150         cnt_max *= 10;
151     else if (coverage == SDRAM_TEST_LONG)
152         cnt_max *= 1000;
153
154     puts("SDRAM: Running EMIF Diagnostic Test ...");
155     //status = test_rand_address(cnt_max, addr_bgn, addr_end, seed);
156
157     status = test_address_kg(addr_bgn, addr_end);
158
159     if (status)
160         puts("Passed\n");
161     else
162         puts("Failed\n");
163     return status;
164 }
```

5.5.1.5. 从所有地址写入和读取的实例编码

```

79 int test_address_by( unsigned int addr_bgn, unsigned int addr_end)
80 {
81     int i;
82     int cnt;
83     unsigned int data_temp[];
84     unsigned int expected_data[];
85     unsigned int addr, read_data;
86     unsigned int num_address;
87     num_address= addr_end/4; //16 BYTE/4
88
89     data_temp[0]=0xFFFFFFFF; //Initial data for walking 0 pattern
90     data_temp[1]=0x00000001; //Initial data for walking 1 pattern
91     data_temp[2]=0x33333333; //Initial data for A->5 switching
92
93     expected_data[0]=0xFFFFFFFF; //Initial data for walking 0 pattern
94     expected_data[1]=0x00000001; //Initial data for walking 1 pattern
95     expected_data[2]=0xAAAAAAAA; //Initial data for A->5 switching
96
97     addr=addr_bgn;
98     printf("\nSTARTED 4096 DATA PATTERN !!!! THIS WILL TAKE SEVERAL MINUTES\n",data_temp[0]);
99     /*write*/
100     for (cnt = 0 ; cnt < num_address ; cnt++) {
101         writel(data_temp[cnt],addr);
102         addr = addr + 4; /* Byte-Addressing */
103         data_temp[cnt]=ROTATE_RIGHT(data_temp[cnt]);
104         #ifdef CONFIG_HW_WATCHDOG
105         WATCHDOG_RESET();
106         #endif
107     }
108     printf("\nSTARTED READING\n");
109     addr=addr_bgn;
110     /*read*/
111     for (cnt = 0 ; cnt < num_address ; cnt++) {
112         read_data=readl(addr);
113         addr = addr + 4; /* Byte-Addressing */
114         #ifdef CONFIG_HW_WATCHDOG
115         WATCHDOG_RESET();
116         #endif
117     }
118
119     if (read_data !=expected_data[cnt]) {
120         printf("!!!!!! FAILED!!!!!! EXPECTED DATA : %8x   READ DATA:%8x\n",expected_data[cnt],read_data);
121         return 0;
122     }
123     expected_data[cnt]=ROTATE_RIGHT(expected_data[cnt]);
124     }
125     return 1;
126 }

```

Number of logical address = number of bytes /4

writel to write to memory

Byte Addressing

readl to read from memory

5.5.1.6. 读/写 Preloader 中的 HPS 寄存器

使用如下功能:

1. writel 写入 HPS 寄存器
 - writel(value, address);
2. readl 从 HPS 寄存器读取
 - readl(address);

例如:

1. write logic "0" to GPO


```
writel(0x00000000,0xFF706010);
```

value → address
2. write logic "1" to GPO


```
writel(0x00000001,0xFF706010);
```
3. Read from GPO & store read back value in variable


```
variable =readl(0xFF706010);
```

address

gpo
Provides a low-latency, low-performance, and simple way to drive general-purpose signals to the FPGA fabric.

Module Instance	Base Address	Register Address
gpoaprrange	0xFF706000	0xFF706010

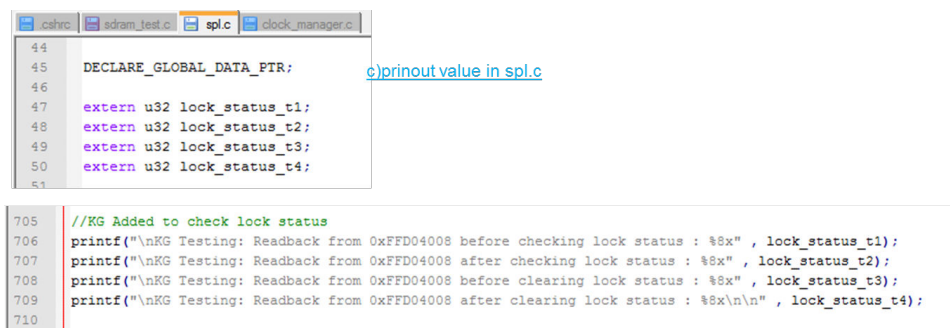
Offset: 0x10
Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value															
HW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value															
HW 0x0															

gpo Fields				
Bit	Name	Description	Access	Reset
31:0	value	Drives 32_0x11:0 with specified value. When read, returns the current value being driven to the FPGA fabric.	RW	0x0

5.5.1.7. 检查 Preloader 中的 HPS PLL Lock Status

- 在 clock_manager.c 中读取 HPS PLL Status Register, 并打印 spl.c 中的内容
 - 定义 clock_manager.c 中的全局变量和 spl.c 中的“extern variable” (外部变量)
 - 尚未使用 UART, 无法打印 clock_manager.c



```
44  
45 DECLARE_GLOBAL_DATA_PTR;  
46  
47 extern u32 lock_status_t1;  
48 extern u32 lock_status_t2;  
49 extern u32 lock_status_t3;  
50 extern u32 lock_status_t4;  
51  
705 //KG Added to check lock status  
706 printf("\nKG Testing: Readback from 0xFFD04008 before checking lock status : %8x", lock_status_t1);  
707 printf("\nKG Testing: Readback from 0xFFD04008 after checking lock status : %8x", lock_status_t2);  
708 printf("\nKG Testing: Readback from 0xFFD04008 before clearing lock status : %8x", lock_status_t3);  
709 printf("\nKG Testing: Readback from 0xFFD04008 after clearing lock status : %8x\n\n", lock_status_t4);  
710
```

5.5.2. 通过 FPGA-to-SDRAM 接口访问 HPS SDRAM

可从 Preloader (SPL/MPL)或 U-Boot 使能 HPS 桥接, 某些情况下也可从 Linux 使能。

注意: 从 SoC EDS 13.1 以及更高版本生成的 Preloaders (SPL)和 U-Boot 包含额外功能和内置功能, 可安全使能 HPS 桥接。

从 Preloader 或 U-Boot 使能 HPS FPGA-to-SDRAM 桥接, 请遵循以下相应步骤。

从 Preloader 使能 HPS-to-FPGA Bridges

Preloader 检查 FPGA 的状态, 如果 FPGA 已配置, 则自动使能 Platform Designer (Standard) 和 BSP 中配置的桥接。Preloader 支持先编程 FPGA, 再运行自动桥接使能测试和代码。

有关更多信息, 请参阅 [GSRD v13.1 - Programming FPGA from HPS](#)。

从 U-Boot 使能 HPS-to-FPGA Bridges

可从 U-boot 命令弹出的 bridge_enable_handoff 命令使能桥接。在该命令将 HPS 和 SDRAM 置于安全状态后, 先适当检查然后才使能全部桥接。

有关更多信息, 请参阅 KDB 解决方案: [How can I enable the FPGA2SDRAM bridge on Cyclone V SoC and Arria V SoC Devices?](#)



A. 支持和文档

A.1. 支持

操作系统供应商提供将操作系统板支持数据包 (BSP) 移植到 Intel SoC 开发套件的技术支持。

Intel 提供 Intel SoC FPGA Embedded Development Suite (SoC EDS) 支持和用于 FPGA 开发的设计工具。SoC EDS 中包含 Arm Development Studio 5 for Intel SoC FPGA Edition Toolkit。

Intel 提供 Intel SoC 开发套件支持。

其他板卡的技术支持由相应板卡供应商或分销商提供。

项目	支持提供方
商业操作系统和工具	操作系统和工具供应商
HardwareLibs (裸机)	Intel
Intel SoC FPGA EDS	Intel
Arm DS-5 for Intel SoC FPGA Edition	Intel
FPGA 设计工具	Intel
Open Source 和 Linux	RocketBoards.org
UBoot	RocketBoards.org

其他有关信息，请参阅如下链接中提供的内容。

相关链接

- [Intel FPGAs](#) 页面中的 **Ecosystem** 选项卡
- [支持](#)
Intel FPGA 产品支持
- [终端市场相关设计](#)
Intel FPGA 产品的参考设计和设计实例
- [Intel FPGA 设计解决方案网络](#)
- [Intel 支持中心](#)
- [Intel FPGA 产品的设计实例](#)
- [开发套件，子卡&编程硬件支持](#)
- [Intel FPGA Wiki](#)

A.2. 软件文档

在当今开源嵌入式软件功能更新特性的推动下，我们的大部分软件文档也随之交由网站管理。

请参阅如下链接中提供的内容了解更多信息。

相关链接

- [RocketBoards.org](#) 文档门户网站
- [Altera Opensource](#)
RocketBoards.org 存储库中的 Linux 相关源代码
- [RocketBoards.org Boards](#)
- [RocketBoards.org Projects](#)
- [Intel SoC FPGA 嵌入式开发套件用户指南](#)
- 下载 [SoC EDS](#)
Intel SoC FPGA Embedded Development Suite 页面中的 **Getting Started** 选项卡

B. 附加信息

B.1. Cyclone V 和 Arria V SoC 器件指南修订历史

文档版本	说明
2020.07.27	更新了如下部分： <ul style="list-style-type: none"> SoC FPGA 嵌入式软件设计指南部分： <ul style="list-style-type: none"> 添加了软件代码管理考量 添加了 SD 卡低功耗模式设计考量 SoC FPGA 中 HPS 部分的设计指南小节： <ul style="list-style-type: none"> 避免寻求 ACP 依赖
2019.07.16	在 <i>Design Guidelines for HPS portion of SoC FPGAs</i> 部分添加了 <i>HPS Address Mirroring</i> 以说明 SoC 中 HPS SDRAM 镜像等级支持。
2018.06.18	更新了早期功耗估算部分，添加了有关 CVSoC L 的信息。
2017.12.22	<ul style="list-style-type: none"> 更新产品名称。 “背景：比较 SoC FPGA 和 SoC FPGA HPS 子系统”章节： <ul style="list-style-type: none"> 删除了 L3 互连的概述和结构框图 删除了 SDRAM 控制器块的结构框图 阐明 FPGA-to-SDRAM 访问的描述。 删除了 HPS-FPGA 系统拓扑结构的详细信息 添加了指南： <ul style="list-style-type: none"> 使用轻量级 HPS-to-FPGA 桥接连接需要 HPS 控制的 IP。 FPGA 存储器不使用轻量级 HPS-to-FPGA 桥接。而是使用 HPS-to-FPGA 桥接。 使用 HPS-to-FPGA 桥接将 FPGA 主控的存储器连接到 HPS。 如果使用连接 HPS-to-FPGA 桥接的存储器进行 HPS 引导，请确保其从地址在 Platform Designer (Standard) 中设置为 0x0。 通过 FPGA-to-HPS 桥接，使用从 FPGA 主接口高速缓存访问 HPS。 使用 FPGA-to-HPS 桥接，从 FPGA 主接口访问 HPS 中高速缓存一致的存储器，外设或片上 RAM。 使用 FPGA-to-SDRAM 端口，从 FPGA 主接口非高速缓存访问 HPS SDRAM。 “SoC FPGA 中 HPS 部分的设计指南”章节： <ul style="list-style-type: none"> 建议采用 Cyclone V HPS-FPGA Bridge Reference Design Example，取代 Cyclone V Datamover Design Example 不建议将 GPIO 用于高速串行接口 添加了指南： <ul style="list-style-type: none"> 使用 Golden System Reference Design (GSRD) 作为松散耦合系统的起点。 使用 Cyclone V HPS-to-FPGA Bridge Design Example 参考设计确定 FPGA 逻辑和 HPS 之间访问的最佳突发长度和数据宽度。 删除的指南： <ul style="list-style-type: none"> Intel 建议使用 Golden System Reference Design (GSRD) 作为松散耦合系统的起点。 Intel 建议使用 Cyclone V HPS-FPGA Bridge Reference Design Example 来优化硬件设计和软件解决方案以达到使用 HPS ARM 处理器时的最高实时应用程序性能。

继续...

文档版本	说明
	<ul style="list-style-type: none">“SoC FPGA 的电路板设计指南” 章节：<ul style="list-style-type: none">HPS 专用 I/O 支持的 RGMII添加了指南：<ul style="list-style-type: none">如果您的设计使用 4-byte 寻址的 QSPI flash，设计电路板时需要确保 HPS 复位时，QSPI flash 也已复位或者已重新启动。如果您的 SPI 外设需要 SPI 主从选择在事务期间保持低电平，请考虑使用 GPIO 作为从选择，或配置 SPI 主选择在事务期间置位从选择。请确保只要 HPS 复位，SD/MMC 卡也随之复位。裸机应用程序中，请避免使用大于 16 MB 的 QSPI flash 器件使用大于 16 MB 的 QSPI 器件时，如果器件支持，请使用 QSPI 扩展 4-byte 寻址命令“SoC FPGA 的嵌入式软件设计指南” 章节：<ul style="list-style-type: none">不再为基于 NAND 引导提供参考 DTB阐明引导支持所需要的 NAND flash 接口类型
2017.02.20	首次发布