

Hello-FPGA CoaXPress 2.0 FPGA HOST IP Core Demo User Manual

目录

Hello-FPGA CoaXPress 2.0 Host FPGA IP Core Demo4

1 说明.....4

2 设备连接5

3 VIVADO FPGA 工程6

4 SDK 工程9

图 1-1 VIVADO 工程目录结构	4
图 1-2 SDK 工程目录结构	5
图 2-1 ZCU102 结构图	6
图 2-2 ZCU102 UART 接口	6
图 3-1 VIVADO 工程	7
图 3-2 CPU 控制器	7
图 3-3 CXP IP 实例化	8
图 3-4 均衡器芯片配置 IP	9
图 4-1 CXP Demo SDK 软件工程目录	10
图 4-2 Debug 配置, 下载并复位	10
图 4-3 串口连接	11
图 4-4 Debug 界面及其串口输出信息	11
表 1-1 LINK 速率配置	5

Hello-FPGA CoaXPress 2.0 Host FPGA IP Core Demo

1 说明

本手册针对 Hello-FPGA 的 CoaXPress 2.0 HOST FPGA IP Core demo 工程，用于演示 IP 的使用方法、配置流程。本文的内容不仅适用于 ZCU102，也同样适用于其他开发板。

Demo 特点功能如下：

- 代码适用于 ZCU102, KCU105, KC705, AXKU040/2, AX7P, AX19P 等使用 Xilinx 芯片作为主控制器的评估板；
- 使用 VIVADO 2019.1 及其 SDK；
- VIVADO 工程使用 block design 形式提供；
- LINK 配置为 1 个相机，4 个 LINK，设备发现阶段使用 0x38 配置，即 3.125Gbps，设备采集阶段使用 0x38/0x58 配置，即 12.5Gbps，如果相机不支持对应速率，请修改代码后进行测试；
- 使用 MicroBlaze 作为 CPU 控制器，软件代码使用 SDK 进行开发、调试；
- IP 使用网表形式提供，参数无法修改，如需不同 LINK 配置，请联系 Info@hello-fpga 或其它 Hello-FPGA 工程师
- 范例代码对相机的基本寄存器完成配置，如果需要配置相机的更多参数，用户需要根据示例自行添加，通常来说需要从相机厂商或者自行读取 xml 配置文件，然后根据 xml 描述的寄存器地址对相机进行更丰富的配置。Hello-FPGA 也提供增值服务，帮助用户快速完成指定应用。

文件列表：

ZCU102 顶层文件夹

 cxp_host 示例 FPGA 及其软件驱动

 IPs 工程依赖的 IP 文件，除 CXP IP 外，其余均以源码提供

下图展示了 cxp_host 内部的目录结构，直接使用 VIVADO 2019.1 打开 *.xpr 工程文件即可。








	cxp_host.ip_user_files	2023/7/4 22:50	文件夹	
	cxp_host.runs	2023/7/4 22:50	文件夹	
	cxp_host.sdk	2023/7/4 22:50	文件夹	
	cxp_host.sim	2023/7/4 22:50	文件夹	
	cxp_host.srds	2023/7/4 22:50	文件夹	
	cxp_host.tcl	2023/7/4 13:16	TCL 文件	30 KB
	cxp_host.xpr	2023/7/4 23:05	Vivado Project Fi...	64 KB

图 1-1 VIVADO 工程目录结构

下图展示了 *.sdk 内部目录结构，直接使用 Xilinx SDK 打开即可。

.metadata	2023/6/30 20:01	文件夹	
cxp_host_wrapper_hw_platform_0	2023/7/4 13:17	文件夹	
hello_cxp	2023/6/30 20:06	文件夹	
hello_cxp_bsp	2023/7/4 13:18	文件夹	
RemoteSystemsTempFiles	2023/6/30 20:01	文件夹	
webtalk	2023/6/30 20:43	文件夹	
cxp_host_wrapper.hdf	2023/7/4 9:54	HDF 文件	1,350 KB
SDK.log	2023/7/4 13:18	文本文档	6 KB

图 1-2 SDK 工程目录结构

表 1-1 LINK 速率配置

速率配置	Downlink 速率	最大速率
0x28	1.250 Gbps	1.000 Gbps
0x30	2.500 Gbps	2.000 Gbps
0x38	3.125 Gbps	2.500 Gbps
0x40	5.000 Gbps	4.000 Gbps
0x48	6.250 Gbps	5.000 Gbps
0x50 ⁽¹⁾	10.000 Gbps	8.000 Gbps
0x58 ⁽¹⁾	12.500 Gbps	10.000 Gbp

2 设备连接

我们先用 ZCU102 为例进行说明：

- 1、Camera, camera 请按照厂商要求连接电源；
- 2、Camera 与 CXP HOST FMC 子卡连接，请注意 LINK 序号一一对应，使用 CXP 同轴线缆完成连接；
- 3、CXP HOST FMC 与 ZCU102 FMC2 (板上丝印 HPC0)连接，连接好后请使用螺丝进行固定；
- 4、ZCU102 UART 串口与 JTAG 需要连接到主计算机，其中 JTAG 负责下载 FPGA bitstream 与 debug，UART 负责将 demo 软件打印信息输出，demo 实际使用的是 PL 端的 UART，应当连接 UART 的 interface 2，串口速率 115200。

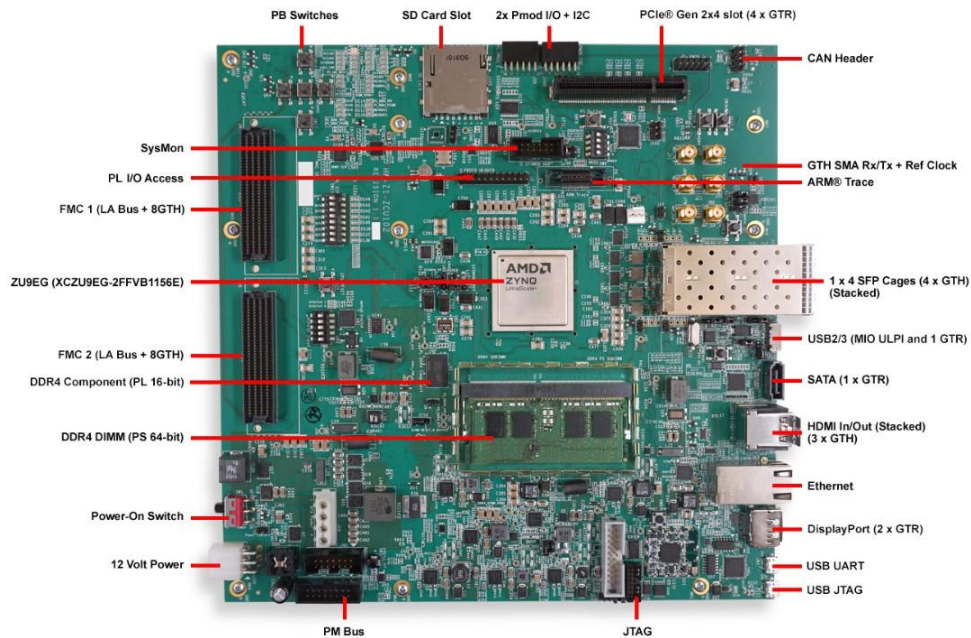


图 2-1 ZCU102 结构图



图 2-2 ZCU102 UART 接口

其它开发板连接方法类似，其中 UART 均使用 PL 端 UART。

3 VIVADO FPGA 工程

使用 VIVADO 2019.1 打开，如果使用其它版本，可以自行升级。

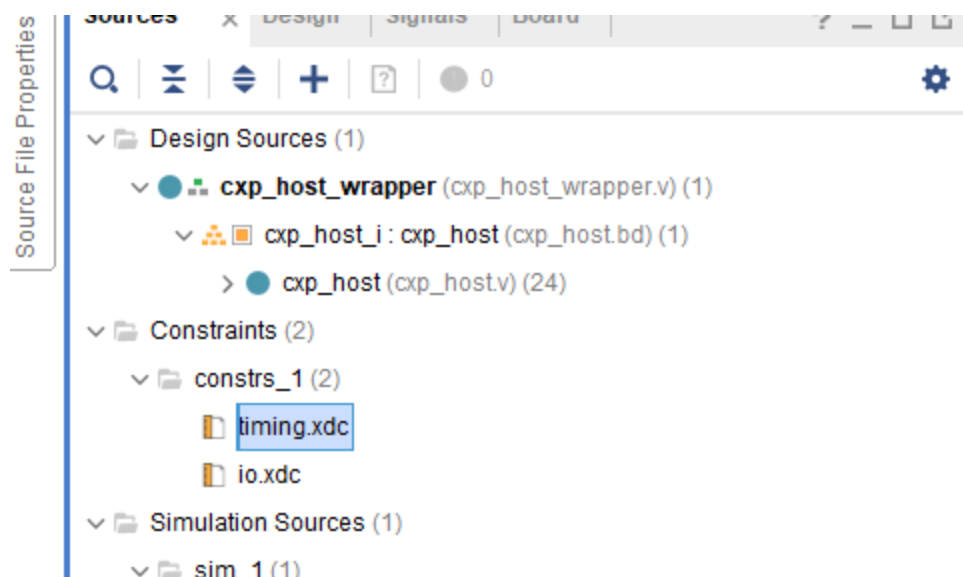


图 3-1 VIVADO 工程

如下图所示，CPU 使用 Xilinx 的 FPGA 软核 MicroBlaze，没有使用 ZYNQ 的 PS 硬核，目的是为了简化 Demo 工程，同时适应不同 FPGA 平台，增强一致性。CPU 与外设之间均通过 AXI 总线进行连接。



图 3-2 CPU 控制器

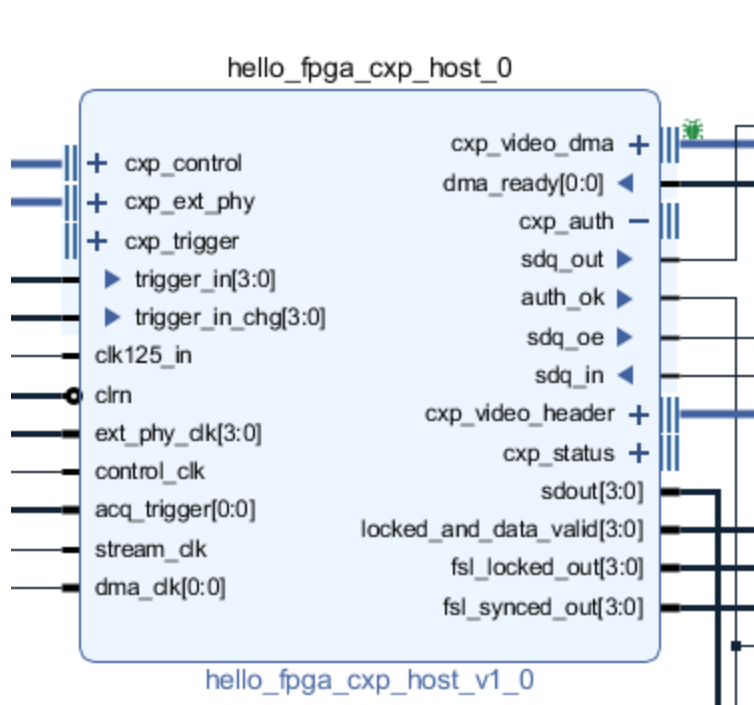


图 3-3 CXP IP 实例化

Demo 使用外部 PHY，使用 AXI lite interface 完成速率的动态配置。

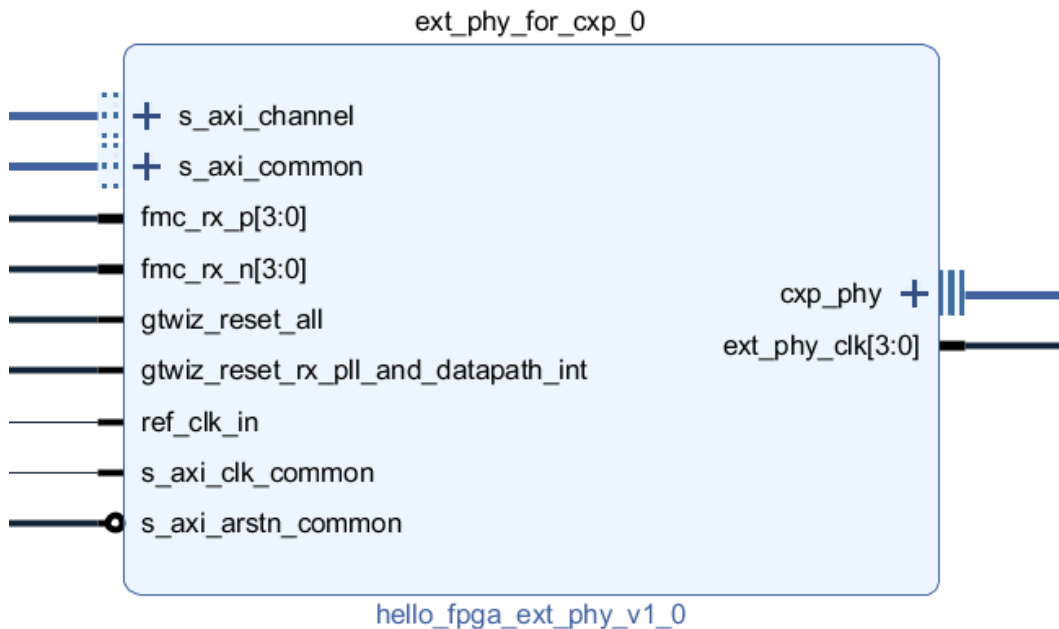


图 3-4 外部 PHY 接口，用 axi lite 完成速率配置

下图为均衡器配置 IP，请注意，该模块仅仅在使用 Microm 均衡器器件才需要，如果使用 Microchip 是不需要的，示例工程也不提供对应实现。均衡器是 FMC 接口板上的一组芯片，用于均衡高速 downlink 信号，不同速率会有不同的参数配置，具体配置逻辑请参考软件 demo。

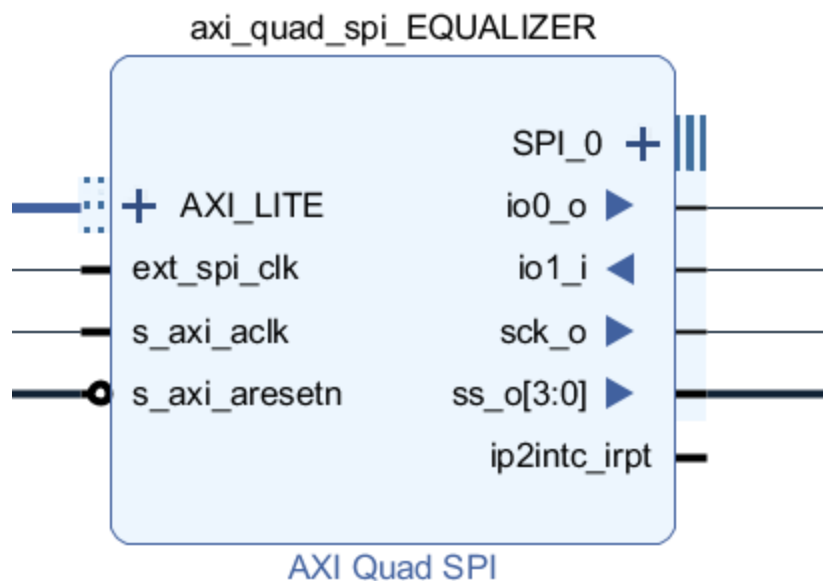


图 3-5 均衡器芯片配置 IP

按照正常流程编译即可。如果代码有修改，改动后需要将硬件信息导出到 SDK，并 LAUNCH SDK 或者其它方式打开 SDK 重新编译软件代码。

4 SDK 工程

使用 2019.1 打开，如果使用 Vitis，请自行导入代码。

下图为裸机 SDK 软件目录，其中 hello_cxp_bsp 和 cxp_host_wrapper_hw_platform 均为系统自动生成 bsp 包，测试 hello_cxp 应用程序即可。

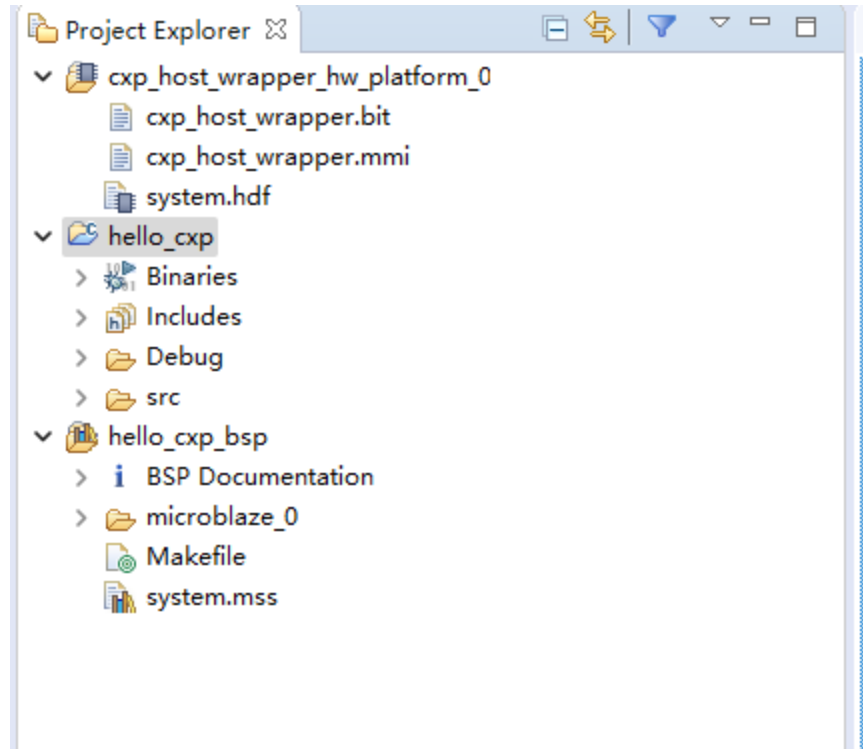


图 4-1 CXP Demo SDK 软件工程目录

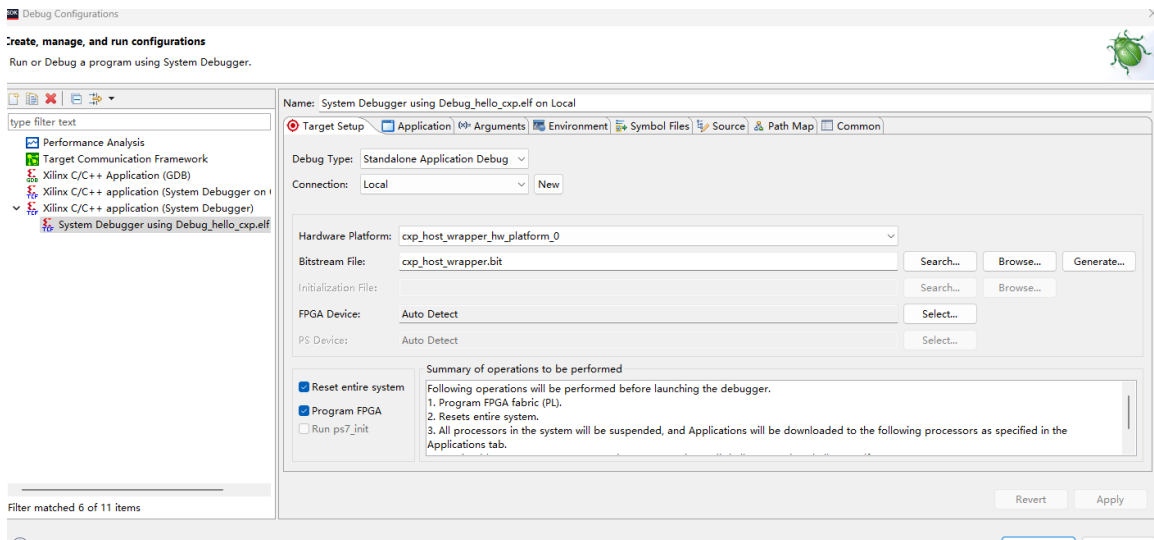


图 4-2 Debug 配置，下载并复位

程序会将打印信息输出到 UART 串口，demo 使用了 PL 端的 UART，对应 interface 接口的 interface2。

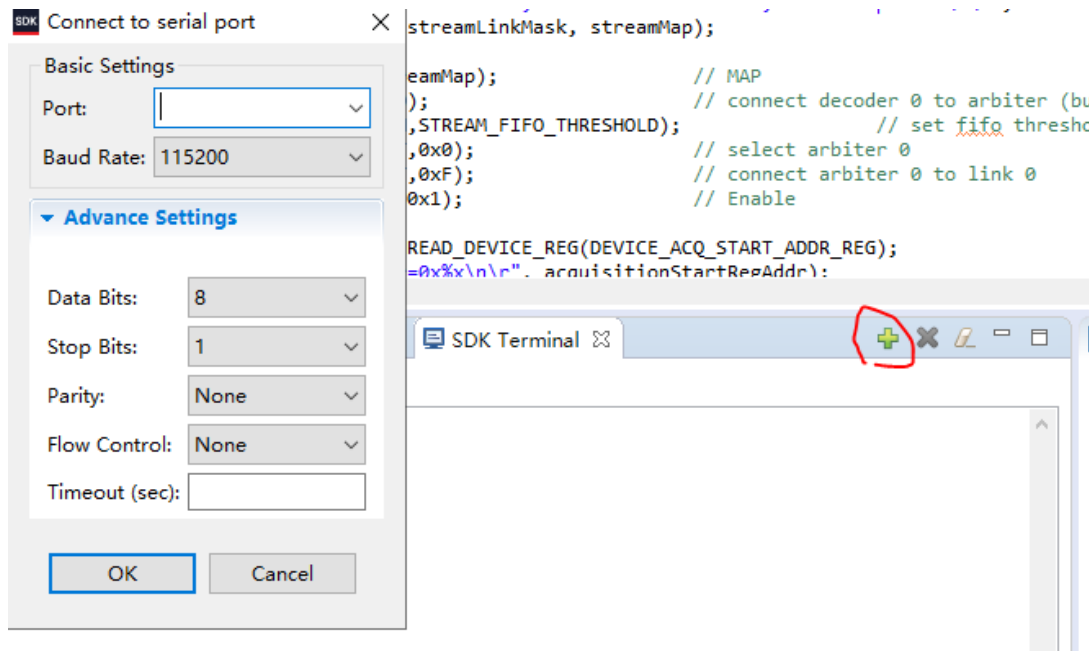


图 4-3 串口连接

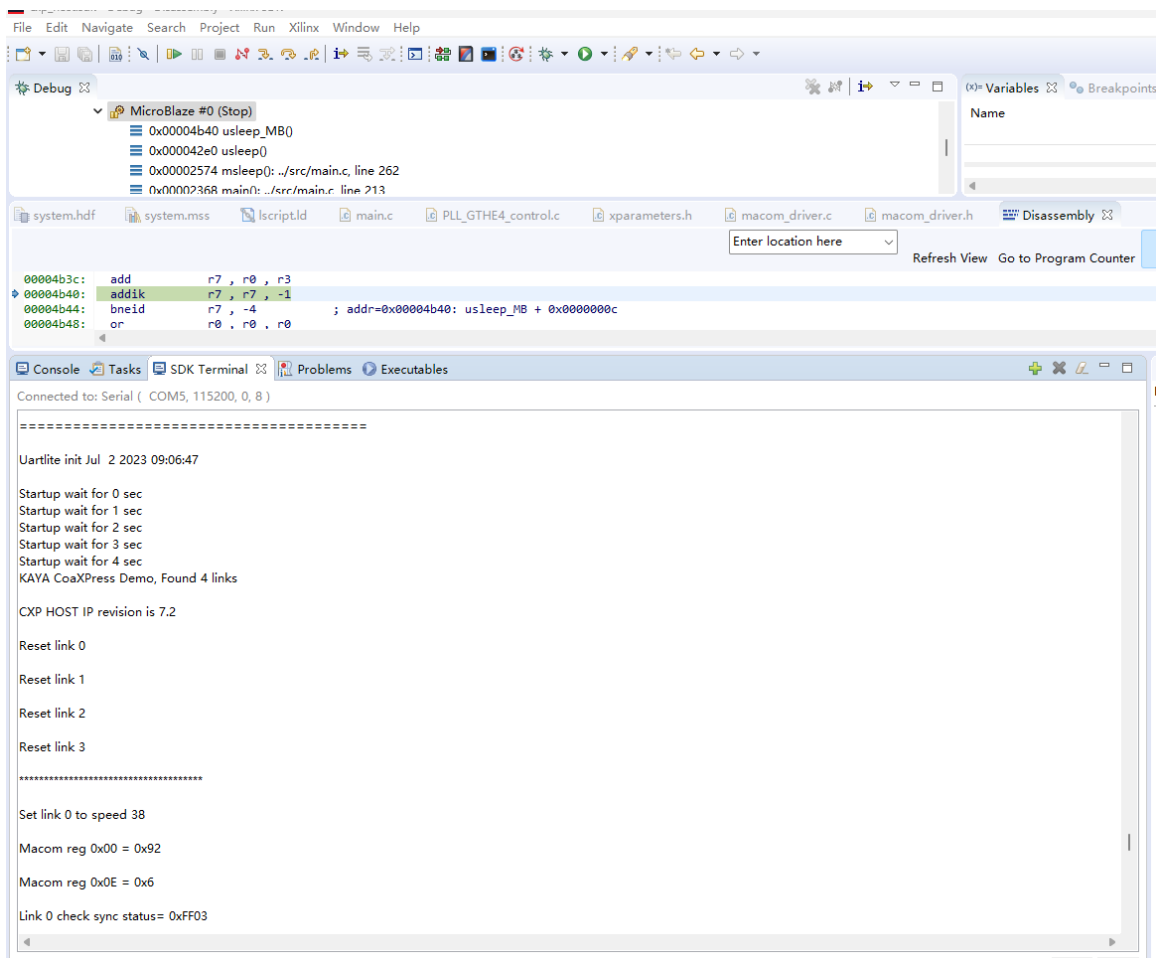


图 4-4 Debug 界面及其串口输出信息

FAQ

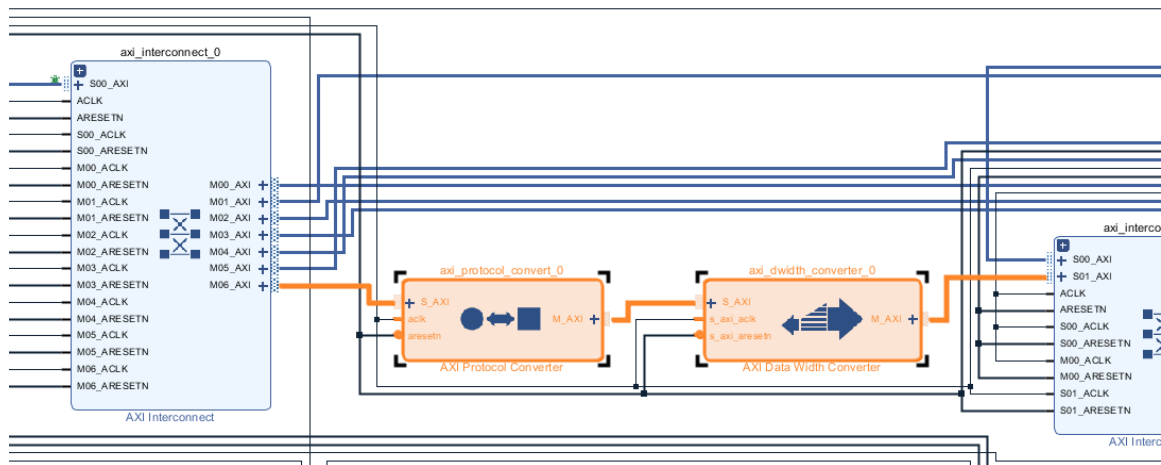
DEMO 目的是提供相机链接，读写示例，本身无法做到很晚上，请仔细阅读 IP 手册。这里对一些常见问题做一些说明。

1: 客户在 demo 基础上增加 Microblaze 外设，典型为 DDR 外设后，运行 SDK 代码出现卡死现象。

原因: `cxp_bursting_axi2ava` 模块不够健壮，在特定情形下发生了总线读数据没有回应的问题导致了卡死。

现象: 客户增加 AXI MIG DDR 控制器模块作为 Microblaze 的标准 AXI 外设，由于 MIG 模块的数据位宽是 512，所以在 Microblaze 对应的 AXI Interconnect 模块内部自动做了位宽转换，这个位宽转换导致 `cxp_bursting_axi2ava` 出现了问题。

解决: 把带来位宽转换的外设部分逻辑在设计中主动添加，不要让系统自动增加(AXI Interconnect 模块会自动根据外设位宽、协议情况增加内部模块，很多时候自动处理会带来冗余设计)。修改后如下图所示:



2: 我们现在换一个相机测试，一直 link 不上，这个版本是不是不支持?

在 LINK 之前，采集端和相机需要协商 LINK 速度，CoaXPress 标准定义了 2 个发现速度，1.25Gbps 和 3.125Gbps，demo 默认使用 3.125Gbps 发现速度，如果发现 LINK 不成功，可以自行修改发现速度。

高速链接 (downlink) 速度选择。

0x28: 通道速率 1.250 Gbps
 0x30: 通道速率 2.500 Gbps
 0x38: 通道速率 3.125 Gbps
 0x40: 通道速率 5.000 Gbps
 0x48: 通道速率 6.250 Gbps
 0x50: 通道速率 10.000 Gbps
 0x58: 通道速率 12.500 Gbps

具体修改 `const uint32_t discoverySpeed[] = { DISCOVERY_SPEED_3,DISCOVERY_SPEED_2};`

3: demo 是 4Lane 设计, 实际只接了 2 条 lane, 发现数据丢失 1 半

“触发是周期是 20us 使用 cyp 12.5 2x(每个像素点是 10bit) 问题是出现 dam 传输数据现丢包问题每两个 dam sol 之间会少一个 dam sol。”

根据 IP 手册, streamLinkMask 寄存器定义了解码器与物理通道的数据连接, demo 默认设置为 0xf, 如果实际只接了 lane 0, lane1 ,那么 streamLinkMask 应该为 0x3;