Nuclei™ N200 系列处理器内核配套 FPGA 实现

版权声明

联系我们

若您有任何疑问，请通过电子邮件 support@nucleisys.com 联系芯来科技。

修订历史

版本号	修订日期	修订的章节	修订的内容
1.0	2019/5/15	N/A	1. 初始版本

目录

图片清单

1. N200 内核配套 SoC

对于 N200 系列处理器内核而言，还需要为其配备原型 SoC 才能展示一个典型 SoC 的功能。Nuclei N200 系列内核搭配完整的原型 SoC 平台，请参见《Nuclei_N200 系列配套 SoC 介绍》了解更多 SoC 的介绍与信息。

N200 系列内核配套 SoC 在 n200_rls_pkg 文件包的文件层次结构如下所示（以 N201 内核为例）。

```
n200_rls_pkg
    |----rtl                    // 存放 RTL 的目录
        |----n201               // N201 核和配套原型 SoC 的 RTL 目录
            |----core           // 存放 N201 Core 的 RTL 代码
            |----fab            // 存放配套 SoC 总线 bus fabric 的 RTL 代码
            |----subsys         // 存放配套 SoC 子系统顶层的 RTL 代码
            |----mems           // 存放配套 SoC 的 memory 模块的 RTL 代码
            |----perips         // 存放配套 SoC 外设 peripherals 模块的 RTL 代码
            |----soc            // 存放 SoC 顶层的 RTL 代码
                |----n201_soc_top.v     // SoC 顶层
```

各个主要的代码模块简述如下：

- fab 目录主要实现 SoC 中 ICB Bus Fabric 模块的 Verilog RTL 代码。
- subsys 目录包含了 SoC 的主体顶层模块的 Verilog RTL 代码，其中 n201_subsys_top 是事实上的 SoC 顶层文件，它例化了 Main Domain 模块 （n201_subsys_main.v）。n201_subsys_mems 模块实现了系统存储总线（System Memory Bus）。n201_subsys_perips 模块实现了系统设备总线（System Peripheral Bus）
- mems 目录主要用于存放 memory 模块的 Verilog RTL 代码，由于 Memory 的具体实现依赖于芯片生产加工厂（foundry）譬如 SIMC 或者 TSMC 的 Memory 宏单元，因此本文件夹下的 Verilog RTL 代码仅仅是行为模型。
- perips 目录主要用于存放各种外设（Peripherals）模块的 Verilog RTL 代码，譬如 GPIO，UART，SPI 等。
- SoC 目录主要用于存放 SoC 顶层模块的 Verilog RTL 代码。n201_soc_top.v 是一个简单地顶层 SoC Wrapper 模块，将 n201_subsys_top 进行例化。

为了便于用户能够快速地移植 N200 内核以及配套 MCU 原型 SoC（在本文中将其简称为“MCU SoC”或者“SoC”），芯来科技公司定制了基于 Xilinx FPGA 的专用开发板“Hummingbird EV KIT”（在本文中将其简称为“蜂鸟 FPGA 开发板”）和专用 JTAG 调试器“Hummingbird Debugger KIT”（在本文中将其简称为 “蜂鸟 JTAG 调试器”）。

完整的 FPGA 开发板原型（包括 FPGA 开发板和调试器）如图 1-1 所示。

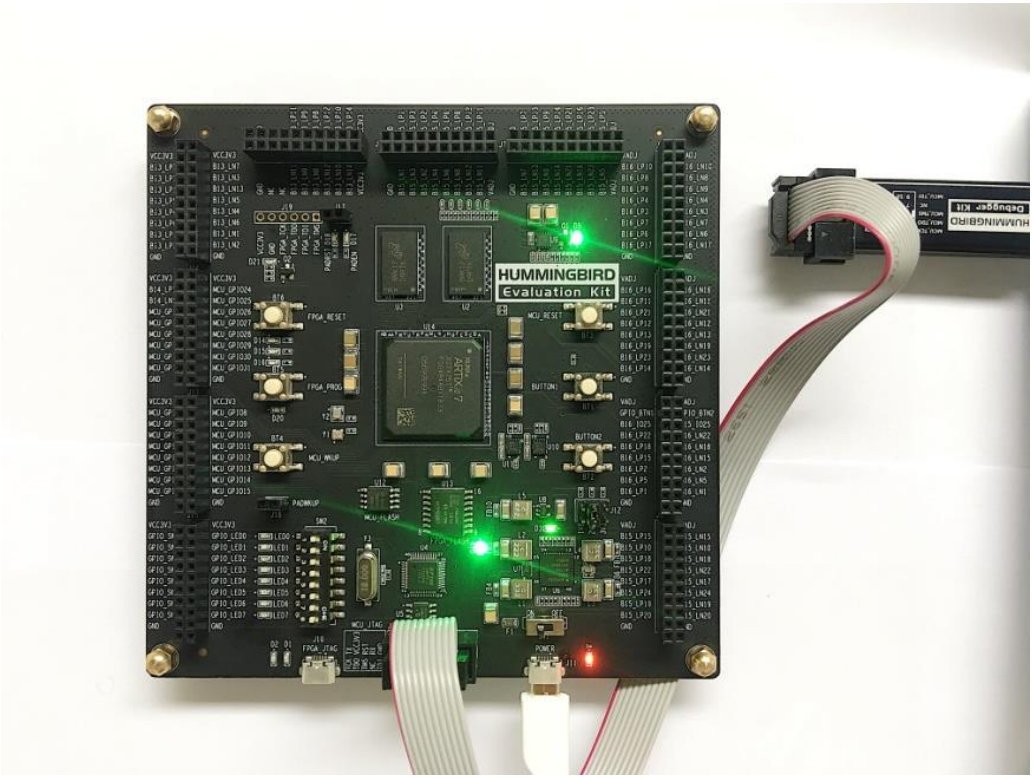

图 1-1 蜂鸟 FPGA 开发板和 JTAG 调试器

后续章节将分别予以详述。

2. 蜂鸟 FPGA 开发板

2.1. 开发板概述

蜂鸟 FPGA 开发板具备如下特点：

- 一板两用，该开发板不仅可以用于一块 FPGA 开发板作为电路设计使用，同时由于其预烧了 N200 配套 SoC（包括 N200 系列内核），因此可以直接作为一块 MCU SoC 原型开发板进行嵌入式软件开发。即：
  - 对于不懂 FPGA 软件开发的用户完全无需做任何的操作，该开发板会预先烧写 N200 Core 和配套 SoC，上电后即可当做一块 MCU 嵌入式开发板来用。
  - 对于了解 FPGA 使用的硬件用户而言，也可以将其当做普通的 FPGA 开发板来烧写普通的 Verilog 电路以进行 FPGA 开发。
    - 用户可以根据自身需求，对 N200 系列处理器内核进行配置，并对其配套的 SoC 进行修改或二次开发。

2.2. 开发板购买途径

用户可以在网页上（https://github.com/SI-RISCV/e200_opensource/tree/master/boards）了解此开发板的购买渠道。

2.3. 开发板硬件配置

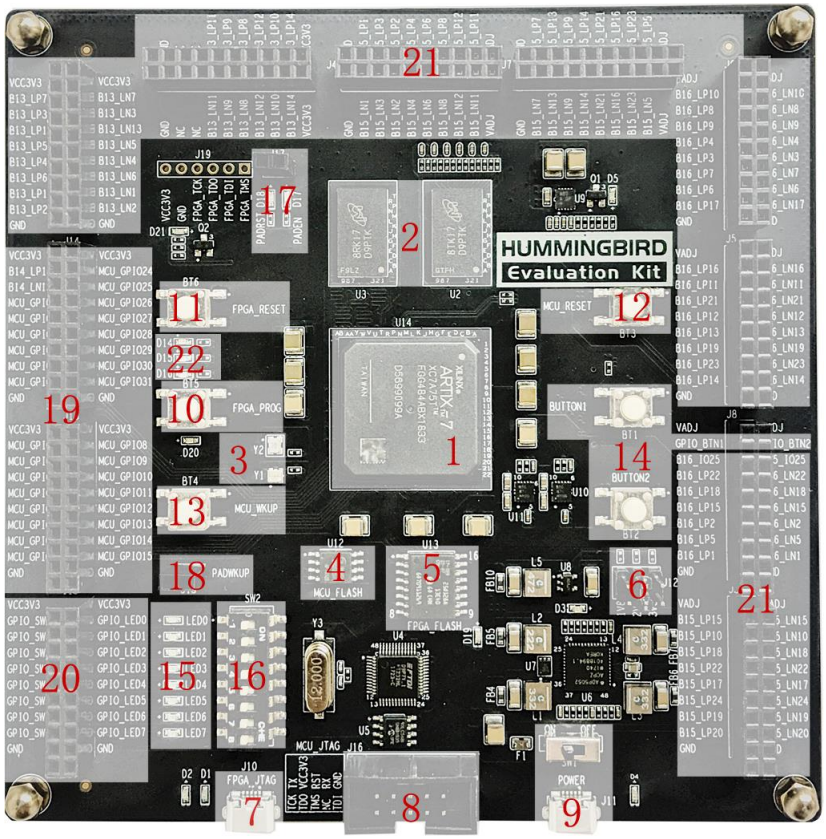

图 2-1 蜂鸟 FPGA 开发板硬件配置

蜂鸟 FPGA 开发板是一款入门级 Xilinx FPGA 开发板，如图 2-1 蜂鸟 FPGA 开发板硬件配置所示，该 FPGA 开发板的硬件特性如下：

- 1：FPGA 主芯片为 Xilinx XC7A75T-2FGG484I （工业级）。
- 2：配备两颗 MT41K128M16JT-125K DDR III 颗粒。
- 3：板载双晶振设计，Y1 为 100MHz 主时钟，Y2 为 32.768K RTC 时钟。
- 4：独立的 MCU_FLASH 芯片，此 Flash 用于存储 RISC-V 内核运行的程序文件。
- 5：独立的 FPGA_FLASH 芯片，此 Flash 用于存储 FPGA 烧录的 mcs 格式比特流文件。
  - 熟悉 Vivado 和 Xilinx FPGA 使用的用户应该了解，bitstream 文件烧录到 FPGA 中去之后 FPGA 不能掉电，因为一旦掉电之后 FPGA 烧录的内容即丢失，需要重新使用 Vivado 的 Hardware Manager 进行烧录方能使用。为了方便用户使用，Xilinx 的 FPGA 开发板可以将需要烧录的内容写入开发板上的 Flash 中（以 mcs 格式），然后在每次

FPGA 上电之后通过硬件电路自动将需要烧录的内容从外部的 Flash 中读出并烧录到 FPGA 之中（该过程非常的快，不影响用户使用）。由于 Flash 是非易失性的内存，具有掉电后仍可保存的特性，因此意味着将需要烧录的内容写入 Flash 后，每次掉电后无需使用 Hardware Manager 人工重新烧录（而是硬件电路快速自动完成），即等效于，FPGA 上电即可使用。

- 6：配备了 FPGA_GPIO 电平选择跳线接口，可以选择 1.8V\2.5V\3.3V 接口电平。
- 7：配备板载的 FPGA USB JTAG 下载器，用于对 FPGA 进行比特流的烧写。
- 8：配备了蜂鸟 JTAG 调试器接口（MCU_JTAG），用于对 RISC-V 内核进行程序的烧写。
- 9：配备单独直流 5V 供电，并设有电源开关。
- 10：除了上电自动对 FPGA 重新进行烧录外，用户还可以通过强行按 FPGA 开发板上的“FPGA_PROG”按键触发硬件电路使用此 Flash 中的内容对 FPGA 重新进行烧录。
- 11：配备独立的 FPGA_RESET 按键，用户可用此按键作为 FPGA 的复位按键。
- 12：配备独立的 MCU_RESET 按键，用户可用此按键作为 RISC-V 内核的复位按键。
- 13：配备独立的 MCU_WKUP 按键。（预留，N200 系列内核及其 SoC 未使用）
- 14：用户按键 1 和 2 及其引出接口。
- 15：用户 8 位 LED。
- 16：用户 8 位拨码开关。
- 17：PADRST 和 PADEN 信号接口及状态指示 LED。（预留，N200 系列内核及其 SoC 未使用）
- 18：PADWKUP 信号接口及状态指示 LED。（预留，N200 系列内核及其 SoC 未使用）
- 19：独立的 32 个 MCU_GPIO，为了便于此开发板直接作为 MCU 原型嵌入式开发板使用，将 N200 系列配套 SoC 的顶层引脚直接连到开发板上，并配有明显的丝印标注（MCU_GPIOxx）。请参见第 2.5 节了解详细介绍。
- 20：用户 LED 及用户拨码开关的引出接口。
- 21：配备多达 126 个引出的 FPGA_GPIO，用于用户自定义扩展使用。
- 22：用户可编程 RGB LED 灯。
- 配备多个电源状态及功能指示 LED 灯。

2.4. 开发板原理图

该开发板的原理图完全开源，保存于 https://github.com/SI-RISCV/e200_opensource/tree/master/boards/hbirdkit 网址，请用户自行查阅。

2.5. 开发板的 MCU 部分

为了便于此开发板直接作为 MCU 原型嵌入式开发板使用，将“N200 系列配套 SoC”的顶层引脚直接连到开发板上，并配有明显的丝印标注，详细描述如图 2-1 所示，其要点如下：

- FPGA 预先烧写成为“N200 系列配套 SoC”
  - 为了实现 MCU 的功能，在 FPGA 开发板上专门配备了一颗 SPI Nor Flash 用于存储 MCU 的软件程序。
  - MCU SoC 的两个输入时钟输入分别按照如下方式产生：
    - 低速的实时时钟直接由 FPGA 开发板上的 32.768KHz 时钟源输入。
    - 高速时钟由 FPGA 开发板上的 100MHz 时钟经过 FPGA 内部 PLL 降频而得（16MHz）。
- 将 SoC 的相关输入输出管脚明确的做到 FPGA 开发板上，并且用印刷字体明确的表明端口号。
  - 有关此 SoC 的输入输出管脚列表的详细信息请参见《Nuclei_N200 系列配套 SoC 介绍》。
  - 注意：所有的管脚都只是映射到 FPGA 内部的普通端口（双向 IO）上，然后通过 FPGA Project 通过设置端口映射把 FPGA 端口映射到这些外部预定义的开发板引脚。有关 FPGA Project 的详细信息请参见第 4 章。
- MCU_JTAG 插槽的 JTAG 引脚连接到 SoC 的四根 JTAG 引脚上；用于调试的 UART 引脚连接到 SoC 的 GPIO_16 和 GPIO_17 上，为 SoC 的 UARTO 所使用。
- MCU_RESET 按键连到 SoC 的输入管脚 AON_ERST_N 上，用于复位 SoC。

- 三色 LED 灯分别显示了 SoC 三根 GPIO 的电平值，Red 与 GPIO 19 连接，Green 与 GPIO 21 连接，Blue 与 GPIO22 连接。

2.6. 开发板的常规功能部分

为了便于此开发板作为常规 FPGA 开发板使用，配备分离的用户拨码开关和用户 LED 灯，如图 2-2 蜂鸟 FPGA 开发板的拨码开关和 LED 灯以及跳线示例中所示：

- 用户 8 位拨码开关，可以分别设置为高电平或低电平。(图 2-2 蜂鸟 FPGA 开发板的拨码开关和 LED 灯以及跳线示例中将 SW0 和 SW1，分别连接到 MCU_GPIO6 和 MCU_GPIO7)。

- 用户 8 位 LED，可以直观的显示对应端口的电平状态。(图 2-2 蜂鸟 FPGA 开发板的拨码开关和 LED 灯以及跳线示例中将 MCU_GPIO11 和 MCU_GPIO12,分别连接到 LED6 和 LED7)

  用户拨码开关会连接到旁边的一排插槽上，从而用户可以使用跳线控制其它的输入或者输出信号。同时，LED 也连接到该插槽上，用于显示对应端口电平（点亮或是熄灭）。

- 用户按键 1 和用户按键 2，分别使用杜邦线连接到 MCU_GPIO30 和 MCU_GPIO31 端口。（如图 2-2 蜂鸟 FPGA 开发板的拨码开关和 LED 灯以及跳线示例所示）。

  - 注意：此处用户拨码开关和用户 LED 并没有被直连到 FPGA 的管脚上，用户可以自由的进行跳线使其控制开发板上的其它信号。如图 2-2 蜂鸟 FPGA 开发板的拨码开关和 LED 灯以及跳线示例中，用户可以通过杜邦线跳线，将用户拨码开关或用户 LED 与 MCU SoC 的 GPIO 端口连接，相当于通过拨码开关来产生 MCU GPIO 的输入，通过 LED 来反应输出电平状态，从而可以编程构建灵活直观的用户 Demo；用户也可以将用户按键 1 和用户按键 2，使用杜邦线跳线，连接到 MCU_GPIO30 和 31 端口上，用于 demo_eclic 程序的演示。

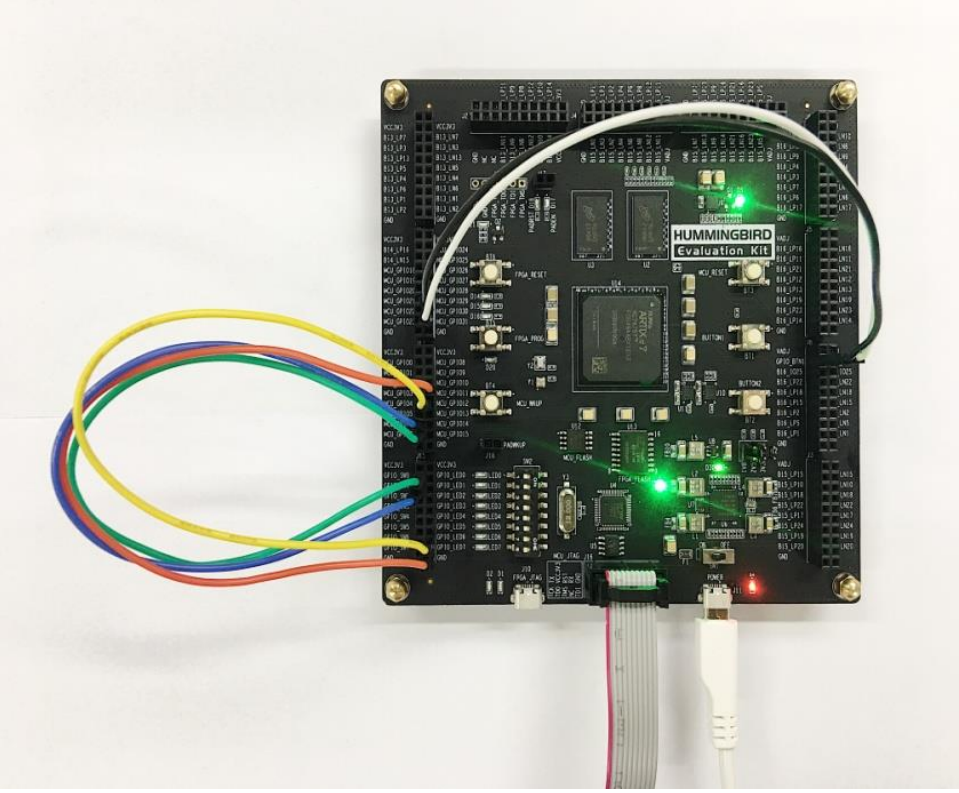

图 2-2 蜂鸟 FPGA 开发板的拨码开关和 LED 灯以及跳线示例

2.7. 烧写 N200 系列内核至开发板

有关如何烧写 N200（包括 SoC 和处理器内核）至此 FPGA 开发板的具体操作步骤，请参见第 4.2 节。

2.8. 使用开发板进行软件开发与调试

有关如何使用 FPGA 开发板进行软件开发与调试的具体操作步骤，请参见第 5 章。

3. 蜂鸟 JTAG 调试器

3.1. 调试器概述

图 3-1 蜂鸟 JTAG 调试器

为了便于用户能够快速地移植 N200 系列内核，芯来科技公司定制了专用的 JTAG 调试器，如图 3-1 所示，该调试器具有如下特性：

- 调试器的一端为普通 USB 接口，便于直接将其插入主机 PC 的 USB 接口，另一端为标准的 4 线 JTAG 接口和 2 线 UART 接口。
- 调试器具备 USB 转 JTAG 功能，通过标准的 4 线 JTAG 接口与蜂鸟 FPGA 开发板连接。由于 N200 系列内核支持标准的 JTAG 接口，通过此接口可以程序下载或进行交互式调试。
- 调试器具备 UART 转 USB 功能，通过标准的 2 线 UART 接口与蜂鸟 FPGA 开发板连接。由于嵌入式系统往往没有配备显示屏，因此常用 UART 口连接主机 PC 的 COM 口（或者将 UART 转换为 USB 后连接主机 PC 的 USB 口）进行调试，这样便可以将嵌入式系统中的 printf 函数重定向打印至主机的显示屏。参见《Nuclei_N200 系列 SDK 使用说明》了解更多详情。

3.2. 调试器购买途径

用户可以网页上（https://github.com/SI-RISCV/e200_opensource/tree/master/boards）了解此 JTAG 调试器的购买渠道。

3.3. 调试器与开发板连接

蜂鸟 JTAG 调试器与蜂鸟 FPGA 开发板的连接方法如图 3-2 中所示。

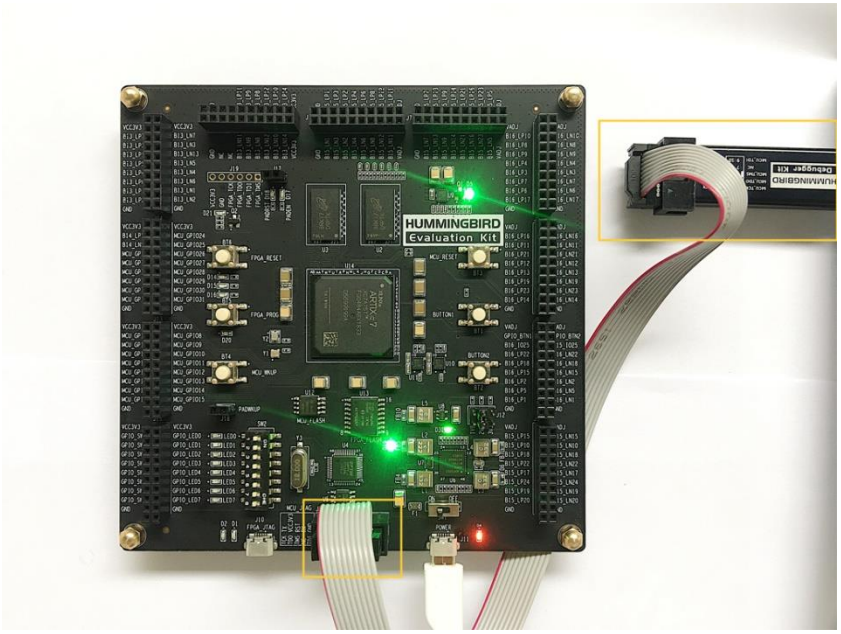

图 3-2 蜂鸟 JTAG 调试器与 PC 和开发板连接

3.4. 使用调试器进行软件下载与调试

有关如何使用 JTAG 调试器进行软件下载与调试的具体操作步骤，请参见第 5 章。

4. 搭建 FPGA 原型平台

芯来科技公司为 N200 系列配套 SoC 定制了专用的 FPGA 原型开发板和 JTAG 调试器，基于该 FPGA 开发板，非常易于使用 N200 系列配套 SoC 搭建完整的原型平台与示例。

FPGA 原型主要分为两部分：FPGA 开发板，和 JTAG 调试器。接下来章节分别予以介绍。完整的 FPGA 开发板原型（包括 FPGA 开发板和调试器）如图 4-1 所示。

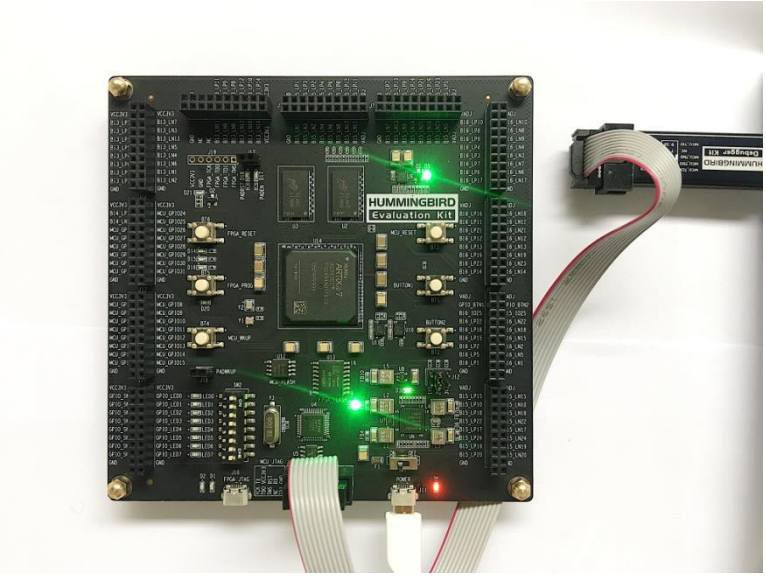

图 4-1 FPGA 开发板原型（包括 JTAG 调试器）

4.1. FPGA 开发板和项目介绍

N200 系列内核专用 FPGA 开发板是一款低成本的入门级 Xilinx FPGA 开发板，如图 4-2 所示。该开发板不仅可以用于一块 FPGA 开发板作为电路设计使用，同时由于其预烧了 N200 系列配套 SoC（包括 N200 系列内核），因此其可以直接作为一块 MCU SoC 原型开发板进行嵌入式软件开发。

有关此 FPGA 开发板的详细介绍和购买渠道请参见网页 https://github.com/SI-RISCV/e200_opensource/tree/master/board。

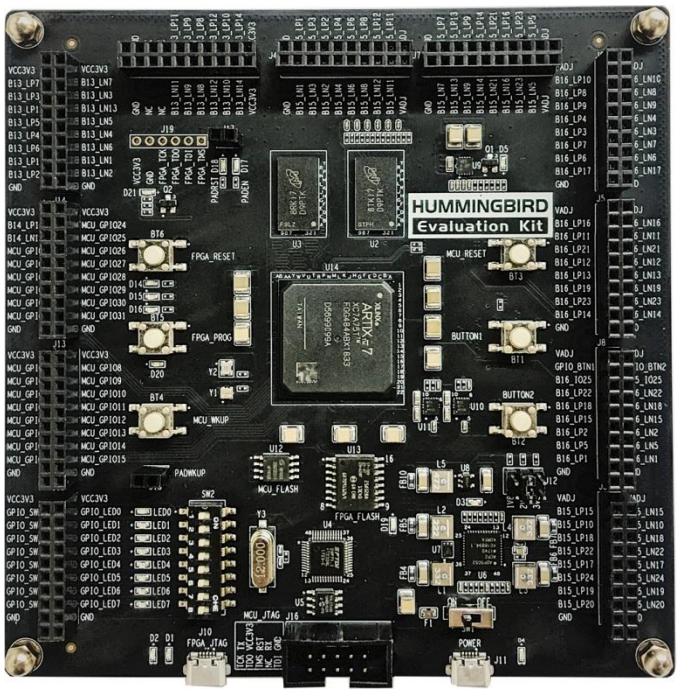

图 4-2 N200 系列内核专用 FPGA 开发板

N200 配套 FPGA 项目在 n200_rls_pkg 文件包的文件层次结构如下所示：

```
n200_rls_pkg
    |----rtl                                  // 存放 RTL 的目录
    |----fpga                                 // 存放 FPGA 项目的目录
        |----hbirdkit                         // 蜂鸟 FPGA 开发板的项目文件夹
            |----constrs                      // 存放约束文件的文件夹
                |---- nuclei-master.xdc       // 主约束文件
            |----Makefile                     // Makefile 脚本
            |----script                       // 存放运行 TCL 脚本的文件夹
            |----src                          // 存放 Verilog 源代码的文件夹
                |----system.org               // FPGA 系统的顶层模块
        |----Makefile                         // Makefile 脚本
```

FPGA 项目通过 Makefile（fpga/common.mk 文件）将添加一个特殊的宏 FPGA_SOURCE 至 Core 的宏文件中，如图 4-3 所示。所以最终用于编译 FPGA 比特流的 RTL 源代码必须包含此宏（FPGA_SOURCE）。

```
20 install:
21     mkdir -p ${PWD}/install
22     cp ${PWD}/../rtl/${CORE} ${INSTALL_RTL} -rf
23     cp ${PWD}/artydevkit/src/system.org ${INSTALL_RTL}/system.v -rf
24     sed -i 's/n200/${CORE}/g' ${INSTALL_RTL}/system.v
25     sed -i '1i\`define FPGA_SOURCE\'  ${INSTALL_RTL}/core/${CORE}_defines.v
```

图 4-3 FPGA 项目宏定义文件中添加 FPGA_SOURCE

在 FPGA 的顶层模块（system.org）中除了例化 SoC 的顶层（n201_soc_top）之外，主要是使用 Xilinx 的 I/O Pad 单元例化顶层的 Pad。另外使用 Xilinx 的 MMCM 单元生成时钟。注意：SoC 的 Main Domain 使用的 MMCM 产生的高速时钟连接到 SoC 的 hfextclk，Always-On Domain 使用的是开发板上的低速实时时钟（32.768KHz）。

N200 系列配套 SoC 的顶层 I/O Pad 经过 FPGA 的约束文件（nuclei-master.xdc）进行约束使之连接到 FPGA 芯片外部的引脚上面（譬如将 JTAG I/O 约束到开发板的 MCU_JTAG 插口引脚上）。

4.2. 生成 mcs 文件并烧写

在前文中介绍了 N200 项目的 SoC 整体架构和 Verilog RTL 代码，为了使得该 SoC 能够真正运行在 FPGA 硬件上，需要将其编译成为 bitstream 文件然后烧录到 FPGA 中去。可以使用如下步骤进行编译和烧录（以 N201 内核为例）。

// 步骤一：准备好自己的电脑环境，可以在公司的服务器环境中运行，如果是个人用户，推荐如下配置：
（1）使用 VMware 虚拟机在个人电脑上安装虚拟的 Linux 操作系统。
（2）由于 Linux 操作系统的版本众多，推荐使用 Ubuntu 16.04 版本的 Linux 操作系统
有关如何安装 VMware 以及 Ubuntu 操作系统本文不做介绍，有关 Linux 的基本使用本文
也不做介绍，请用户自行查阅资料学习。

// 步骤二：安装 Xilinx Vivado 软件（包括 XILINX USB JTAG 下载器驱动程序）至此虚拟机 Linux 操作系统中。有关如何安装 Xilinx Vivado 软件和下载器驱动本文不做介绍，请用户自行查阅资料了解。

// 步骤三：将 n200_rls_pkg 解压至本机 Linux 环境中。

// 步骤四：设置需要编译的 Core 的具体型号，使用如下命令：

```
cd n200_rls_pkg/fpga
        // 进入到 n200_rls_pkg 目录文件夹下面的 fpga 目录。

make install CORE=n201 FPGA_NAME=hbirdkit
        // 运行该命令指明需要为 n201 内核进行编译，该命令会在 fpga 目录下生成一个
        // install 子文件夹，在其中放置 Vivado 所需的脚本，且将脚本中的关键字设置为 n201。
```

```
// 步骤五：生成 bitstream 文件或者 mcs 文件（推荐使用 mcs 文件），使用如下命令：

make bit
        // 运行该命令将调用 Vivado 软件对 Verilog RTL 进行编译生成 bitstream 文件
        // 生成的 bitstream 文件名和路径为
        // n200_rls_pkg/fpga/hbirdkit/obj/system.bit
        // 该 bitstream 文件则可以使用 Vivado 软件的 Hardware Manager 功能将
        // system.bit 烧录至 FPGA 中去。

        // 熟悉 Vivado 和 Xilinx FPGA 使用的用户应该了解，bitstream 文件烧录到 FPGA 中
        // 去之后 FPGA 不能掉电，因为一旦掉电之后 FPGA 烧录的内容即丢失，需要重新使用
        // Vivado 的 Hardware Manager 进行烧录方能使用。
        // 为了方便用户使用，FPGA 开发板可以将需要烧录的内容写入开发板上的
        // Flash 中，然后在每次 FPGA 上电之后通过硬件电路自动将需要加载的内容从外部的
        // Flash 中读出并加载到 FPGA 之中（该过程非常的快，不影响用户使用）。由于 Flash
        // 是非易失性的内存，具有掉电后仍可保存的特性，因此意味着将需要载入的内容写入
        // Flash 后，每次掉电后无需使用 Hardware Manager 人工重新烧录（而是硬件电路
        // 快速自动完成），即等效于，FPGA 上电即可使用。


        // 为了能够将烧录 FPGA 的内容写入 Flash 中，需要生成 mcs 文件，使用如下命令：
make mcs
        // 运行该命令将调用 Vivado 软件对 Verilog RTL 进行编译生成 mcs 文件
        // 生成的 mcs 文件名和路径为
        // n200_rls_pkg/fpga/hbirdkit/obj/system.mcs
        // 该 mcs 文件则可以使用 Vivado 软件的 Hardware Manager 功能将
        // system.mcs 烧录至 FPGA 开发板中的 Flash 中去。
```

如何使用 Vivado 的 Hardware Manager 烧写 mcs 文件至 FPGA 开发板上的 Flash 中去，参考如下步骤：

```
// 前提步骤 1：将 FPGA 开发板的“FPGA JTAG”通过 USB 连接线与电脑的 USB 接口连接。开发板的“FPGA JTAG”的位置请参见图 2-1 标注所示。

// 前提步骤 2：将 FPGA 开发板的“ 5V Power 接口”通过 USB 连接线与电脑 USB 接口或者电源插座连接，并将“电源开关”拨至“ON”档位，对 FPGA 开发板进行供电。开发板的“Power 及电源开关”的位置请参见图 2-1 标注所示。

// 前提步骤 3：按照前提步骤 1 和前提步骤 2 中操作，将 2 根 USB 连接线接插完毕，并给开发板上电后，应如图 4-4 所示。


// 步骤一：打开 Vivado 软件。

// 步骤二：打开 Hardware Manager，会自动连接 FPGA 开发板（如果前提步骤 1 操作正确）。如图 4-5 和图 4-6 所示。

// 步骤三：右键 FPGA Device，选择“Add Configuration Memory Device”。如图 4-7 所示。

// 步骤四：选择如下参数的 Flash，如图 4-8 所示：
            Part n25q128-3.3v
            Manufacturer Micron
            Family n25q
            Type spi
            Density 128
```

Width x1 x2 x4

// 步骤五：弹出“Do you want to program the configuration memory device now?”，选择 OK

// 步骤六：在弹出的窗口中的<Configuration file>对话框中选择添加 n200_rls_pkg/fpga/hbirdkit/system.mcs，然后选择 OK，如图 4-7 所示，则开始烧写 Flash，可能会花费几十秒的时间等待。

// 步骤七：一旦烧写 Flash 成功，则可以通过按开发板上的“FPGA_PROG”按键触发硬件电路使用 Flash 中的内容对 FPGA 重新进行烧录。

注意：FPGA 烧写成功之后，则可以无需再连接“FPGA JTAG”的 USB 连接线。

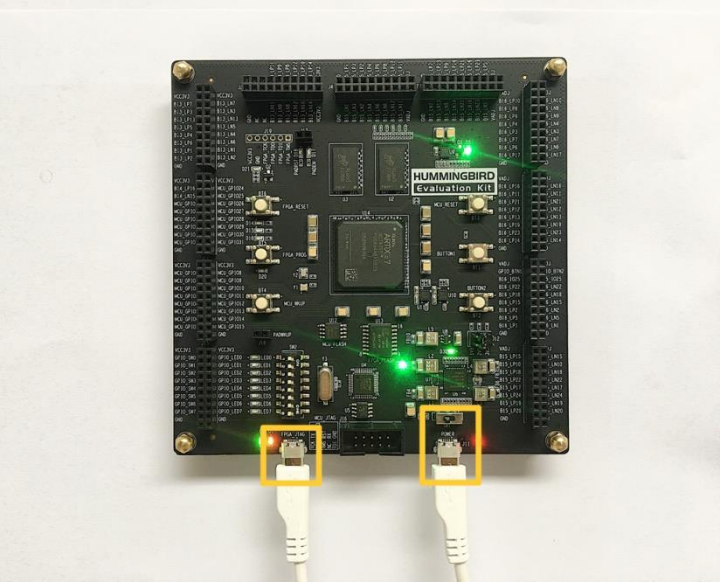

图 4-4 连接 FPGA JTAG 和 5V Power

图 4-5 打开 Vivado Hardware Manager

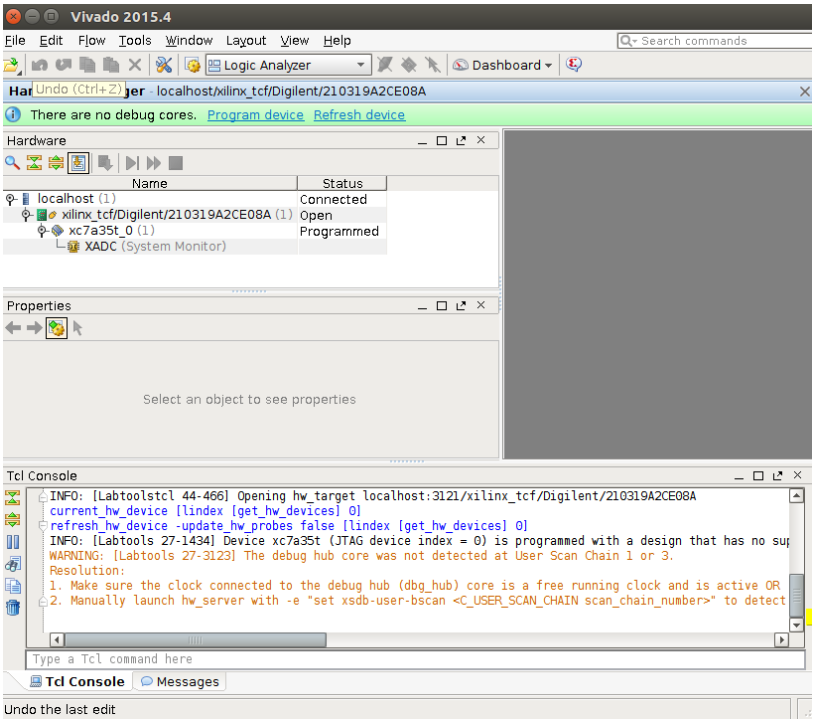

图 4-6 使用 Vivado Hardware Manager 连接 FPGA 开发板

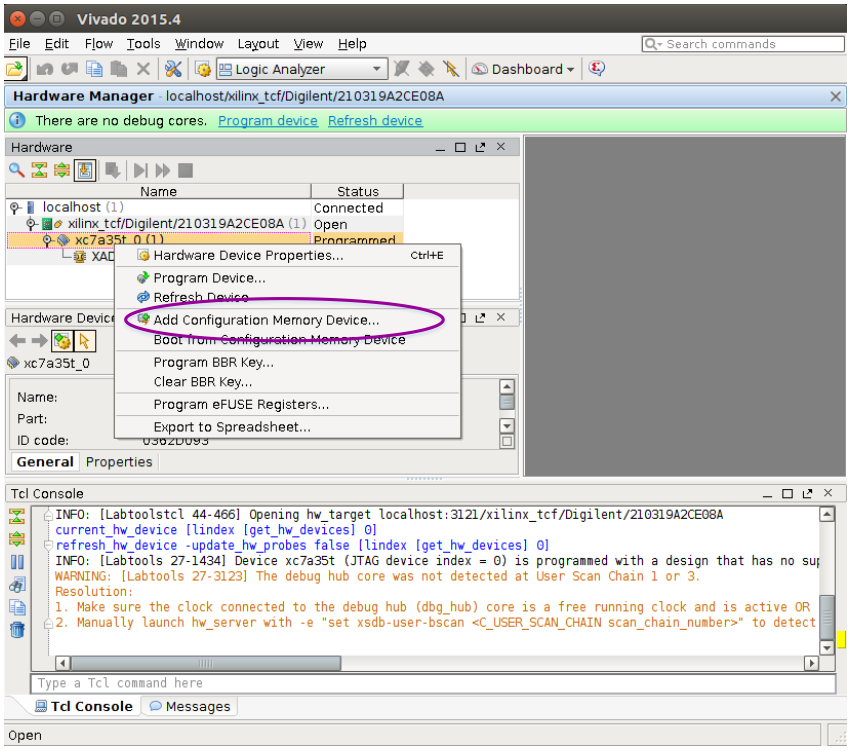

图 4-7 FPGA Device 选择“Add Configuration Memory Device”

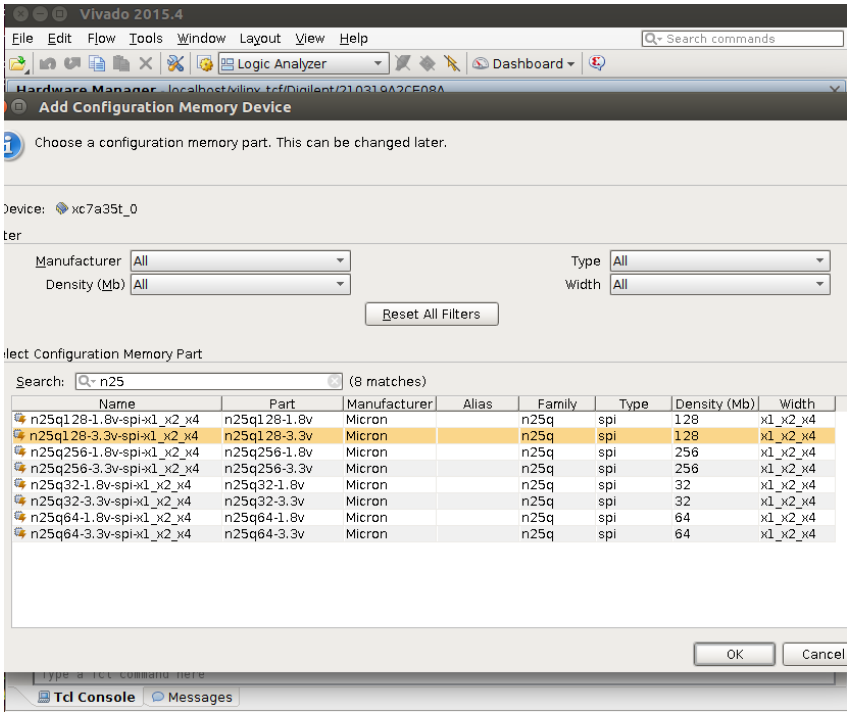

图 4-8 选择 FPGA 配置 Flash 芯片

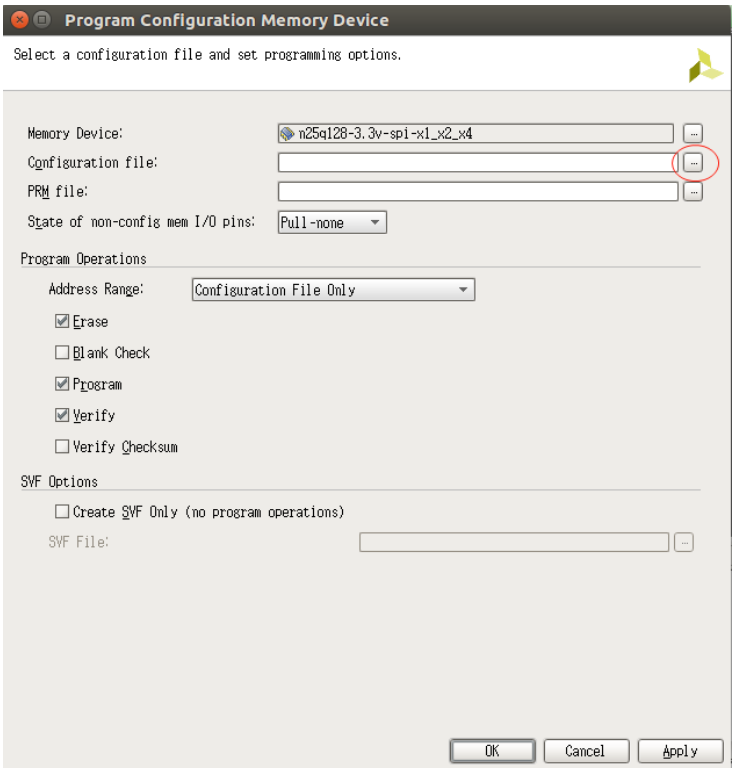

图 4-9 选择需烧写的 mcs 文件

4.3. 连接调试器

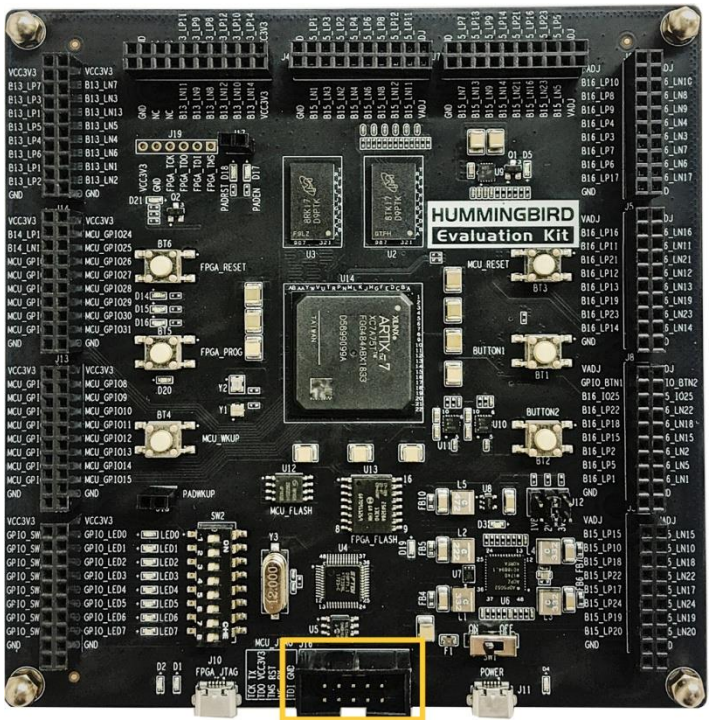

图 4-10 N200 系列内核专用 FPGA 开发板的 MCU_JTAG 插槽

为了支持使用 GDB 进行交互式调试或者通过 GDB 动态下载程序到处理器中运行，需要为 FPGA 原型平台配备一个 JTAG 调试器（JTAG Debugger），N200 系列内核支持通过标准的 JTAG 接口对其进行调试，且 SoC 顶层 JTAG 的 I/O Pad 连接到了 FPGA 芯片的 pin 脚上，而该组 pin 脚在 FPGA 开发板上实际被连接到 MCU_JTAG 插槽上，如图 4-10 中黄色矩形框所示。

芯来科技公司为 N200 系列内核定制了专用的“JTAG 调试器”，如图 4-11 中黄色矩形框所示。

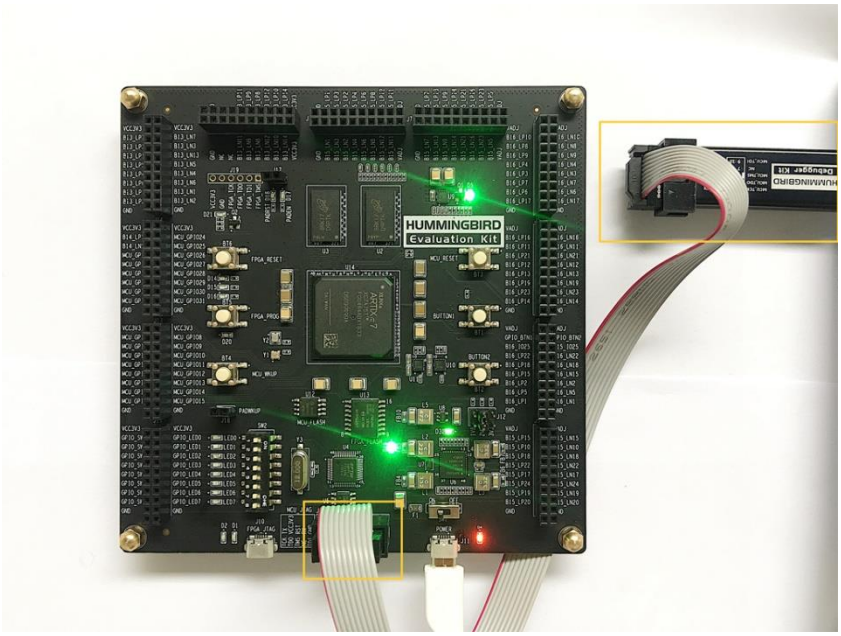

图 4-11 N200 系列内核专用的 JTAG 调试器

有关此 JTAG 调试器的详细介绍请参见《蜂鸟 FPGA 开发板和 JTAG 调试器介绍》。

若想购买此 JTAG 调试器，用户可以通过链接：https://github.com/SI-RISCV/e200_opensource/tree/master/boards 了解此 JTAG 调试器的购买渠道。

由于“JTAG 调试器”将其与上游主机 PC 的 USB 接口连接，因此上游 PC 的 USB 端口需要正确的设置以保证其有正确的权限。以 Ubuntu 16.04 为例，可以使用如下步骤进行配置：

> // 步骤一：准备好自己的电脑环境，可以在公司的服务器环境中运行，如果是个人用户，推荐如下配置：
> （1）使用 VMware 虚拟机在个人电脑上安装虚拟的 Linux 操作系统。
> （2）由于 Linux 操作系统的版本众多，推荐使用 Ubuntu 16.04 版本的 Linux 操作系统
> 有关如何安装 VMware 以及 Ubuntu 操作系统本文不做介绍，有关 Linux 的基本使用本文
> 也不做介绍，请用户自行查阅资料学习。
>
> // 步骤二：使用“JTAG 调试器”将主机 PC 与 FPGA 开发板连接，如图 4-11 中矩形框所示。注意使该 USB 接口被虚拟机的 Linux 系统识别（而非被 Windows 识别），如图 4-12 中圆圈所示，若 USB 图标在虚拟机中显示为高亮，则表明 USB 被虚拟

机中 Linux 系统正确识别（而非被 Windows 识别），将 FPGA 开发板通电。

// 步骤三：使用如下命令查看 USB 设备的状态：

```
lsusb    // 运行该命令后会显示如下信息。
...
Bus 001 Device 029: ID 15ba:002a Olimex Ltd. ARM-USB-TINY-H JTAG interface
```

// 步骤四：使用如下命令设置 udev rules 使得该 USB 设备能够被 plugdev group 所访问：

```
sudo vi /etc/udev/rules.d/99-openocd.rules
         // 用 vi 打开该文件，然后添加以下内容至该文件中，然后保存退出。
# These are for the Olimex Debugger for use with Arty Dev Kit
SUBSYSTEM=="usb", ATTR{idVendor}=="15ba",
ATTR{idProduct}=="002a", MODE="664", GROUP="plugdev"
SUBSYSTEM=="tty", ATTRS{idVendor}=="15ba",
ATTRS{idProduct}=="002a", MODE="664", GROUP="plugdev"
```

// 步骤五：使用如下命令查看该 USB 设备是否属于 plugdev group：

```
ls /dev/ttyUSB*             // 运行该命令后会显示类似如下信息。
/dev/ttyUSB0 /dev/ttyUSB1

ls -l /dev/ttyUSB1     // 运行该命令后会显示类似如下信息。
crw-rw-r-- 1 root plugdev 188, 1 Nov 28 12:53 /dev/ttyUSB1
```

// 步骤六：将您自己的用户名添加到 plugdev group 中：

```
whoami
              // 运行该命令能显示自己的用户名，假设您的用户名显示为 your_user_name
              // 运行如下命令将 your_user_name 添加到 plugdev group 中
sudo usermod -a -G plugdev your_user_name
```

// 步骤七：确认自己的用户是否属于 plugdev group：

```
groups        // 运行该命令后会显示类似如下信息。
... plugdev ...
              // 只要从显示的 groups 中看到 plugdev 则意味着自己的用户属于该组，表示设置成功，如图 4-12 所示
```

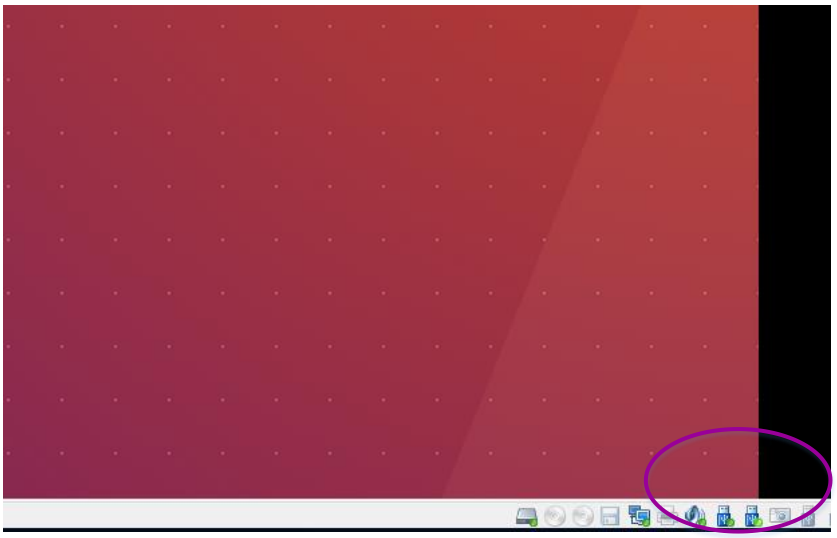

图 4-12 虚拟机识别 USB 设备图标

4.4. FPGA 原型平台总结

至此，将以上论述的加以总结，为了能够搭建完整的 FPGA 原型平台，用户需要做如下硬件的准备：

- 购买一块蜂鸟 FPGA 开发板；
- 购买一个蜂鸟 JTAG 调试器。

用户需要做如下软件的准备：

- 推荐安装 VMware 虚拟机且安装 Linux 操作系统于虚拟机中；
- 安装 Xilinx 公司的 Vivado Design Suite 软件。

在下一章将介绍如何使用烧录后的 FPGA 原型平台，运行真正的嵌入式软件程序示例。

5. 基于 FPGA 平台运行和调试软件示例

5.1. 配套基于 Linux 的 SDK

基于配套的样例 SoC，Nuclei N200 系列处理器内核提供基于 Linux 的软件开发套件（SDK，Software Development Kit）。可以使用此 SDK 在 FPGA 原型平台上进行嵌入式软件的开发，详细介绍请参见参见单独文档《Nuclei_N200 系列 SDK 使用说明》。

5.2. 配套基于 Windows 的 IDE

基于上述配套的样例 SoC，提供基于 Windows 的图形化集成开发环境（IDE，Integrated Development Environment）。可以使用此 IDE 在 FPGA 原型平台上进行嵌入式软件的开发，详细介绍请参见参见单独文档《Nuclei_N200 系列 IDE 使用说明》。