Nuclei® RISC-V FPGA Evaluation Introduction

Nucleisys

Contents

1 Copyright Notice

The product described herein is subject to continuous development and improvement; information herein is given by Nuclei in good faith but without warranties.

This document is intended only to assist the reader in the use of the product. Nuclei do not assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation indirect, incidental, special, exemplary, or consequential damages. incorrect use of the product.

2 Contact Information

Should you have any problems with the information contained herein or any suggestions, please contact Nuclei System Technology by email support@nucleisys.com, or visit "Nuclei User Center" website http://user.nucleisys.com for supports or online discussion.

List of Figures

List of Tables

3 Revision History

Rev.	Revision Date	Revised Section	Revised Content
1.1.0	2023/7/12	N/A	1.First version as the full English
1.2.0	2024/10/25	N/A	1.Fix typo.

4 Nuclei FPGA Evaluation Boards and Debugger Kit Introduction

4.1 Introduction

This document introduce the Nuclei made FPGA evaluation boards, which can be used to prototype the Nuclei evaluation SoC (included RISC-V Core) in FPGA boards; and the Nuclei made JTAG Debugger kit (called *Hummingbird Debugger Kit*), which can be used to debug the RISC-V Core in FPGA prototype or in real chip.

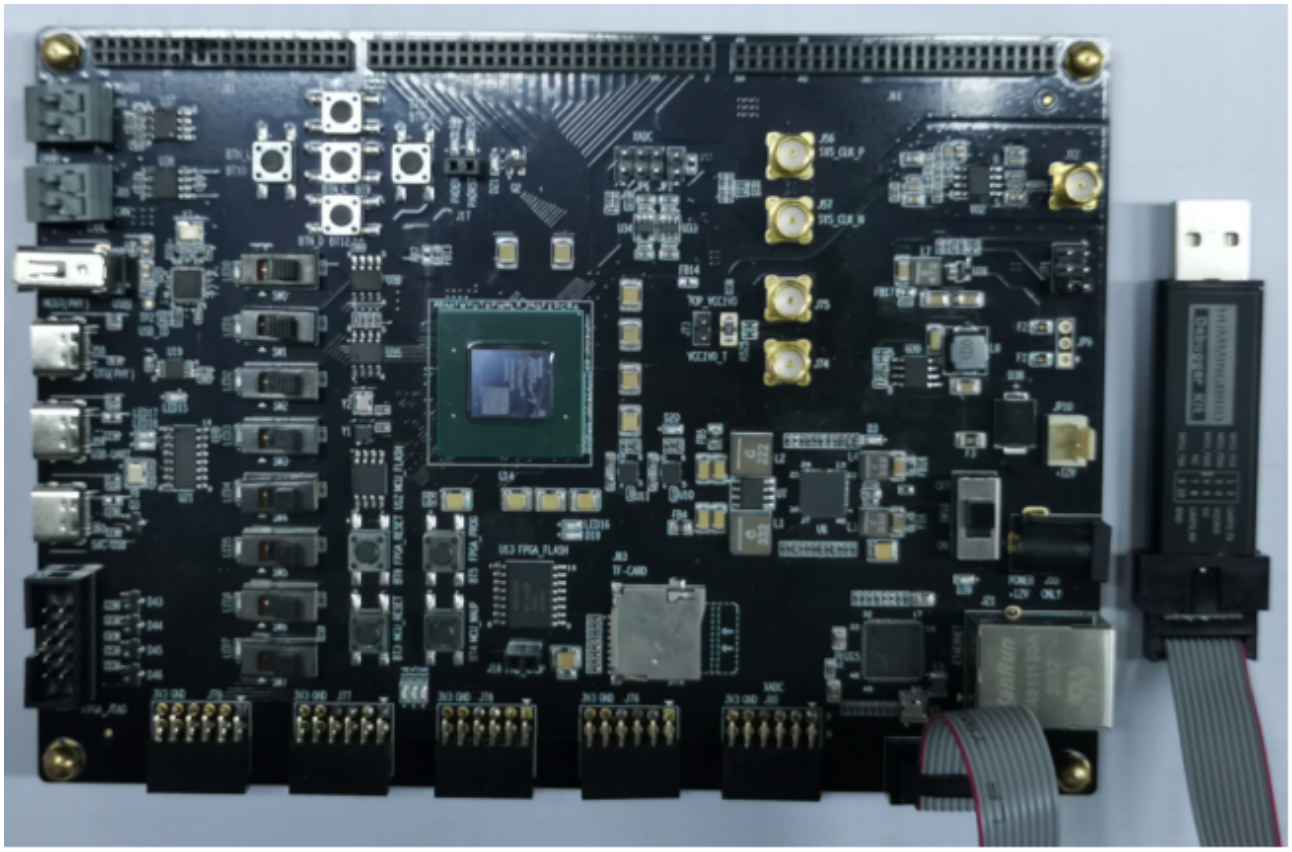

Fig. 4.1: MCU200T Evaluation Kit and Hummingbird Debugger Kit

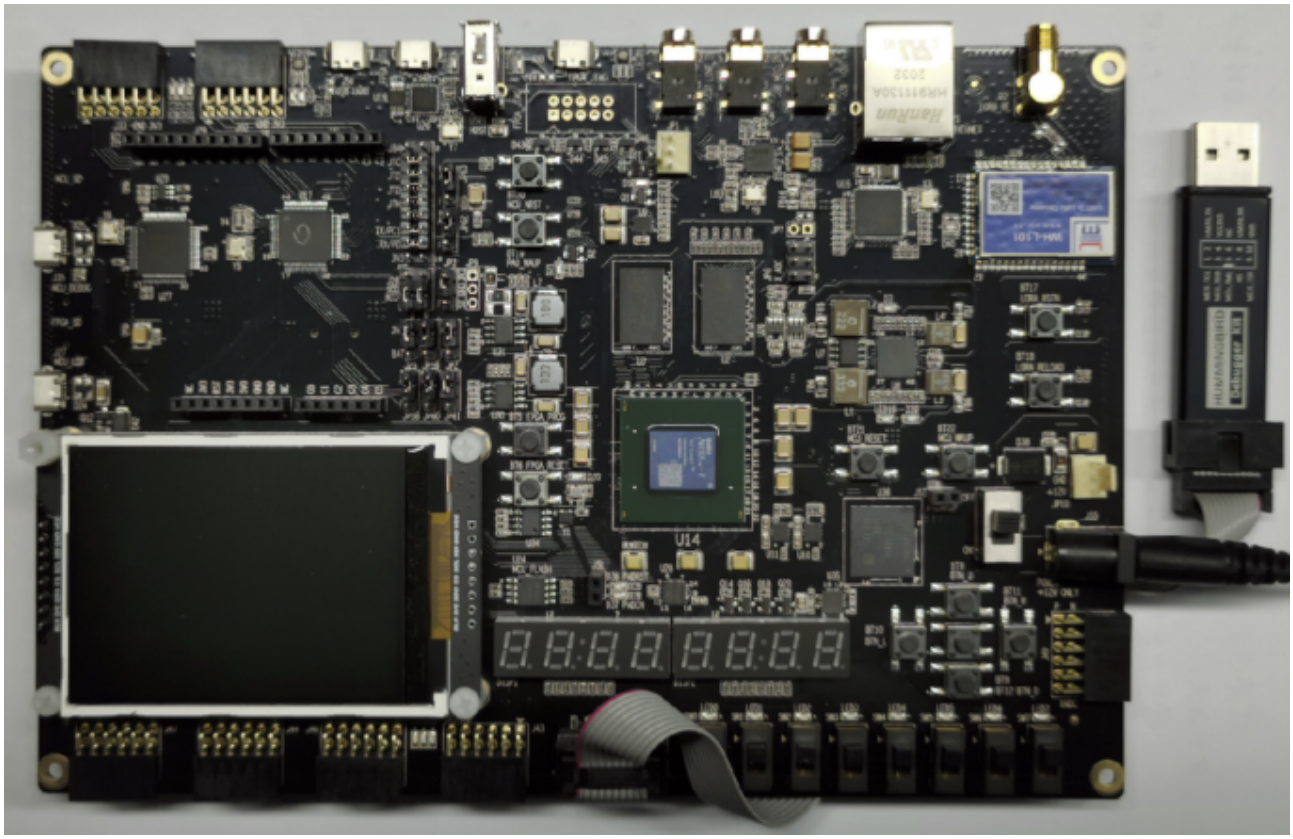

Fig. 4.2: DDR200T Evaluation Kit and Hummingbird Debugger Kit

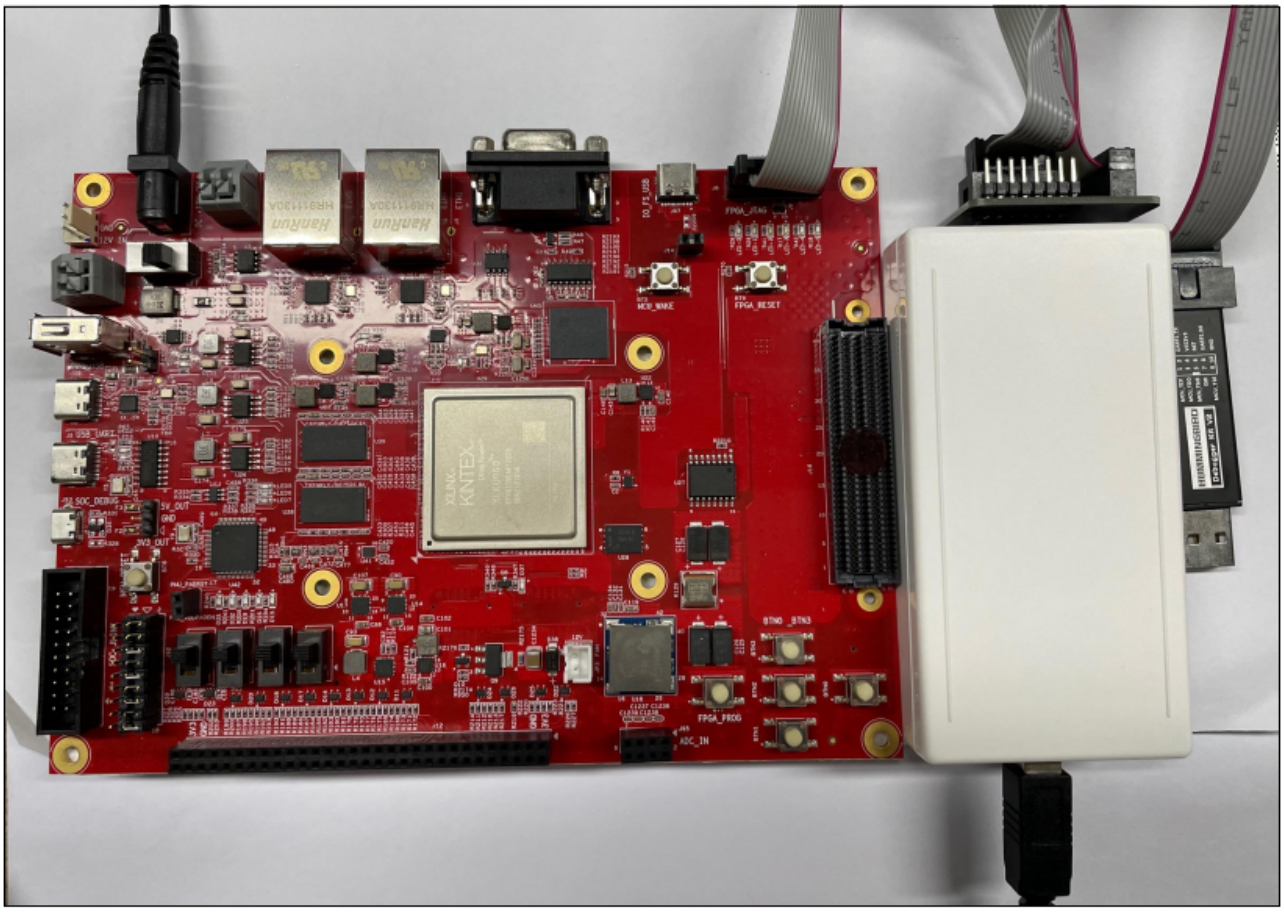

Fig. 4.3: KU060 Evaluation Kit and Hummingbird Debugger Kit

4.2 Hummingbird Debugger Kit

4.2.1 Overview of Debugger Kit

Nuclei have customized a Debugger hardware (called *Hummingbird Debugger Kit*), which can be used to debug the RISC-V core in FPGA prototype or in real chip.

Fig. 4.4: Hummingbird_Debugger_Kit

4.2.2 Features of Debugger Kit

The Hummingbird Debugger Kit, as depicted in Figure 3.4, which have the features as below:

- One end of the Debugger Kit is the USB interface, which can be connected to PC.
- The other end of Debugger Kit is the interface socket, 4 wires of which are for JTAG, and 2 wires are for UART.
- The Debugger Kit is using the FTDI 2232 chip inside it, when it is connected to PC, it will be recognized as two USB devices:
  - One device is to bridge the USB to JTAG, such that the debugger software running on PC (e.g., GDB + OpenOCD) can debug the RISC-V Core through this bridge.
  - Another device is to bridge the UART to USB, such that the SoC's UART's information can also be redirected to PC.

4.2.3 Install the Driver for Debugger Kit

Since the Hummingbird Debugger Kit is connecting PC with USB interface, the PC need to be installed with the drivers to make sure it recognize USB devices.

4.2.3.1 Install the Driver in Linux PC

For the Linux computer, the steps of installing driver for Hummingbird Debugger Kit are as below:

- Step 1: Prepare the PC, it is recommended to have the following configurations.Use the VMware WorkStation to have Linux virtual machine installed, or just use the native Linux OS on your computer. It is recommended to use Ubuntu 16.04 or above version.
- Step 2: Connect the PC with the "Debugger Kit", make sure the USB is really be recognized by the Linux. For example, the USB symbol is highlighted in Ubuntu, as depicted in Figure 3.5.
- Step 3: Use the command to check the USB status:

  lsusb // The example information displayed as below

  ...

  Bus 001 Device 010: ID 0403:6010 Future Technology Devices International, Ltd FT2232xxxx
- Step 4: Use the following command to set udev rules, to make this USB can be accessed by plugdev group

  sudo vi /etc/udev/rules.d/99-openocd.rules

  // Use vi command to edit the file,and add the following lines

  SUBSYSTEM==”usb”, ATTR{idVendor}==”0403”,

  ATTR{idProduct}==”6010”, MODE=”664”, GROUP=”plugdev”

  SUBSYSTEM==”tty”, ATTRS{idVendor}==”0403”,

  ATTRS{idProduct}==”6010”, MODE=”664”, GROUP=”plugdev
- Step 5: Use the following command to check if this USB is belong to plugdev group:

  ls /dev/ttyUSB* // The example information showed as below after this command

  /dev/ttyUSB0 /dev/ttyUSB1

  ls -1 /dev/ttyUSB1 // The example information showed as below after this command

  crw-rw-r– 1 root plugdev 188, 1 Nov 28 12:53 /dev/ttyUSB1
- Step 6: Add your user name into the plugdev group

  whoami

  // Use above command to check your user name, assuming it is your_user_name

  // Use below command to add your_user_name into plugdev group

  sudo usermod -a -G plugdev your_user_name
- Step 7: Double check if your user name is really belong to plugdev group

  groups // The example information showed as below after this command

  ... plugdev ...

  // As long as you can see plugdev in groups, then means it is really belong to.

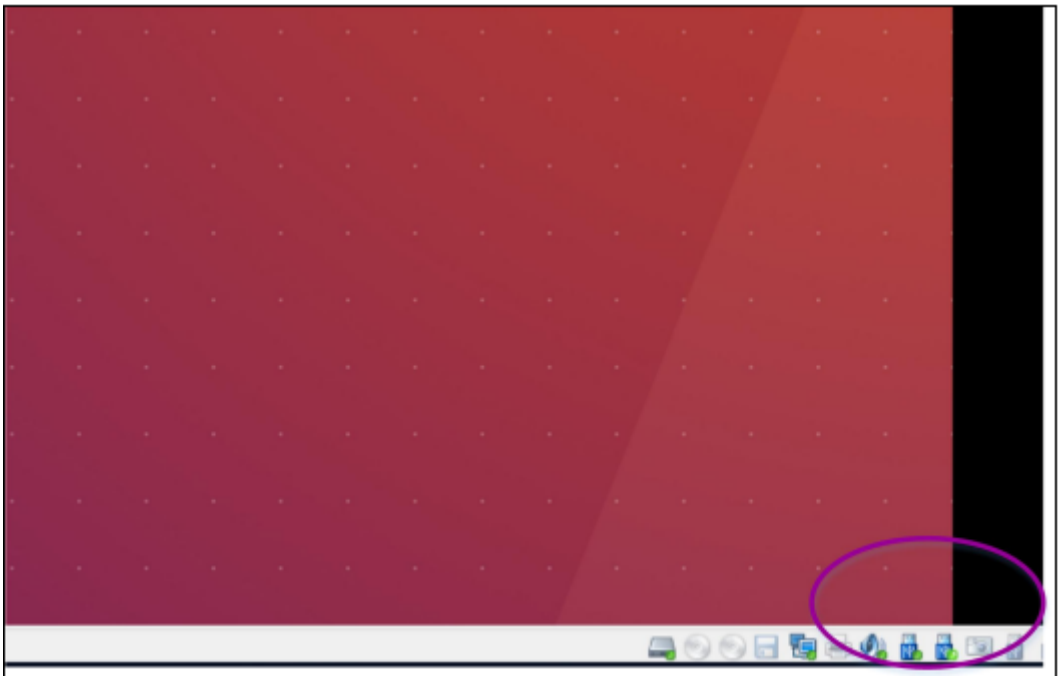

Fig. 4.5: The Ubuntu OS recognized USB

4.2.3.2 Install the Driver in Windows PC

For the Windows computer, the steps of installing driver for Hummingbird Debugger Kit are as below:

- Step 1: Connect the PC with the "Debugger Kit", make sure the USB is really be recognized by the Windows
- Step 2: Download the Hummingbird Debugger Kit's Driver for Windows, from the website: *http://www.nucleisys.com/developboard.php*, as depicted in Figure 3-6.
- Step 3: After downloading the package, double-click the HBird-Driver.exe from it.
- Step 4: Since the Hummingbird Debugger Kit have the functionality that "convert the UART to USB", so if you have connected the "Hummingbird Debugger Kit" with PC and installed the driver successfully, then you will be able to see a USB Serial Port (e.g., COM8) show up in your Windows Device Manager.

Fig. 4.6: Download the Hummingbird Debugger Kit's Driver for Windows

4.3 MCU200T Evaluation FPGA Board

4.3.1 Overview of FPGA board

Nuclei have customized a FPGA evaluation board (called MCU200T Evaluation Kit), which is to make "One board serves two purposes":

- **As the SoC prototype board directly:**
  - If the FPGA have been pre-burned (programmed) with "Nuclei evaluation SoC",this board can be worked as a SoC prototype directly. Since the board has been designed with buttons and extended ports names in line with the SoC GPIO pin name, the embedded software engineers can directly use this board without knowing any FPGA hardware knowledge.
  - To easy the writing, the "Nuclei evaluation SoC" will be shorted as "the SoC" in this document thereby.
- **As the regular FPGA board.**
  - For those hardware engineers who know FPGA hardware knowledge, this board can be used as regular FPGA board, which can be used to burn (program) any Verilog HDL design.

4.3.2 Board Hardware Resources

4.3.2.1 Overall Introduction of Hardware Resources

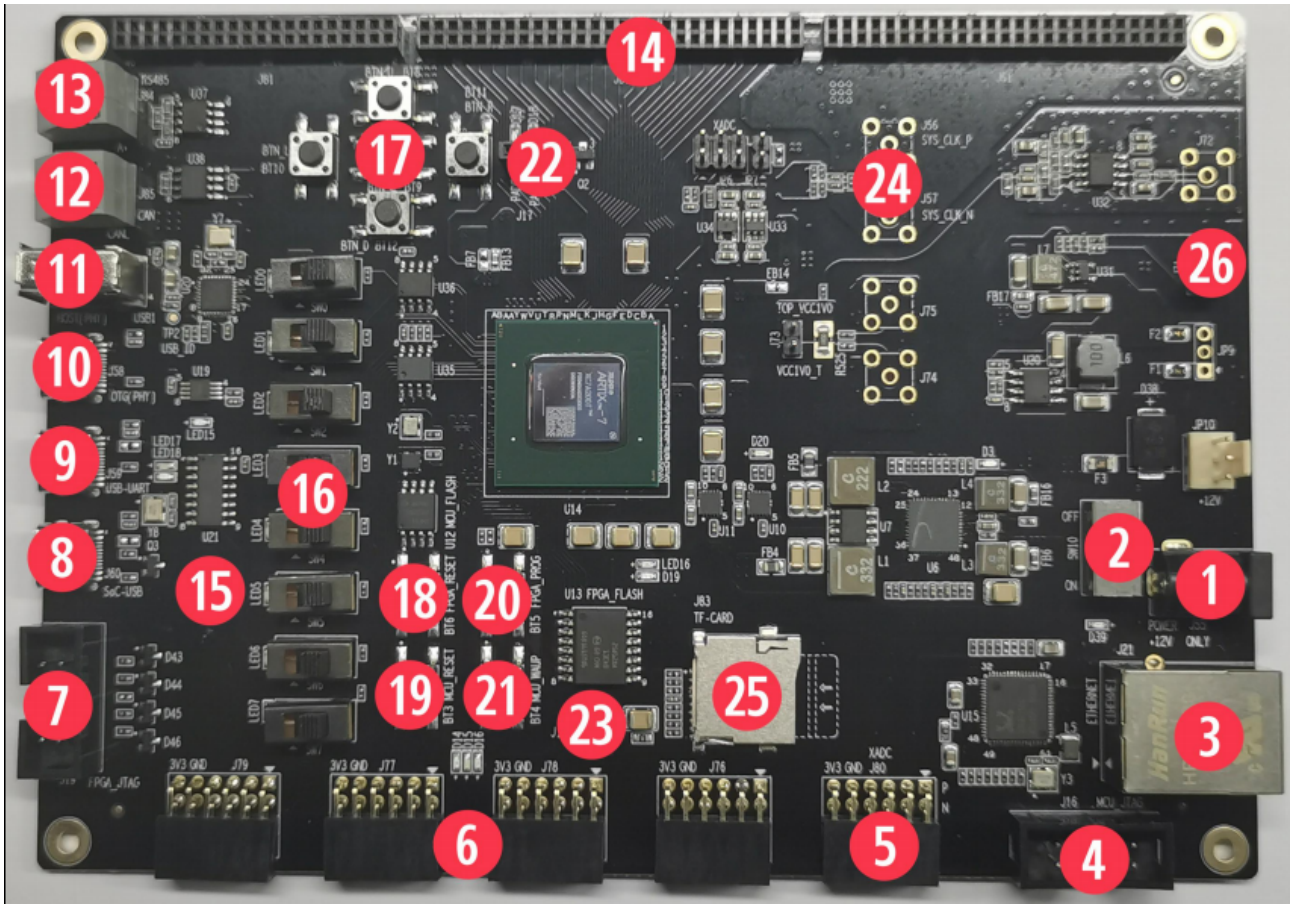

Fig. 4.7: Board Hardware Resources

MCU200T Evaluation Kit is an intermediate-level FPGA board based on Xilinx FPGA, as depicted in Figure 3.7 the hardware resource (marked as with different numbers in above picture) of this board is described as follows.

- The basic hardware resources for FPGA:

  - Xilinx FPGA Chip (XC7A200T-2FBG484I).
  - Two Crystal Oscillators, Y1 is for 100MHz Clock, Y2 is for 32.768K RTC Clock.
  - FPGA_FLASH, independent SPI Flash chip for FPGA to store its Bitstream (MCS format).
- The Xilinx Vivado support the Bitstream to be burned into external SPI Flash with MCS format, the FPGA hardware will automatically re-load the Bitstream from external Flash into the FPGA each time after reset. By this way, FPGA can make sure its Bitstream will not be lost after power-down (since Flash will retain its Bitstream). Please refer to Section 3.3.4 for the steps how to burn MCS format Bitstream into external Flash.
  - The Vadj jumpers, which can be used to configure the voltage level of FPGA_GPIO, as 1.8V, 2.5V, or 3.3V.(Mark: [26])
  - The FPGA_JTAG connector(2.54mm 10PIN), used to connect USB-JTAG Programming cables for programming the Bitstream of the FPGA. (Mark: [7])
  - Separated DC 12V power supply and a power switch. (Mark:[1] & [2])
  - The FPGA_PROG button to force the FPGA to re-load the Bitstream from the external Flash. (Mark:[20])
- The pre-defined hardware resources for the RISC-V Core and the SoC are as below. Please refer to Section 3.3.2.2 for more details of how these resource work.
  - The MCU_FLASH, independent Flash chip for RISC-V core (burned inside FPGA) to store its instruction programs or read-only data.
  - The MCU_JTAG socket, which is the on-board JTAG interface socket for RISC-V Core debugging. (Mark:[4])
  - The MCU_GPIO ports x4 (Compatible with PMOD interface), for 32 GPIO pins of the SoC. (Mark:[6])
  - The FPGA_RESET button for PoR reset of the SoC. (Mark: [18])
  - The MCU_RESET button for System reset of the SoC. (Mark:[19])
- Other hardware resources as regular FPGA board resource:
  - Differential SMA clock input.(Mark:[24])
  - User LEDs x8.(Mark:[15])
  - User Push Buttons x5.(Mark:[17])
  - User DIP Switch (8-position).(Mark:[16])
  - XADC port(Compatible with PMOD interface).(Mark:[5])
  - User CAN port.(Mark:[12])
  - User RS485 port.(Mark:[13])
  - User 64Mb PSRAM.(SPI interface)
  - User IIC EEPROM:24LCo4.
  - FPGA SD Card Slot.(Mark:[25])
  - SoC USB port.(Mark:[8])
  - UART To USB Bridge.(Mark:[9])
  - FPGA USB-OTG port.(Mark:[10]
  - FPGA USB-HOST port.(Mark:[11])
  - 10/100/1000 Mbps Ethernet RJ45 port(GMII). (Mark:[3])
  - FPGA GPIO ports x3(2.0mm).(Mark:[14])
  - The RGB LEDs.

4.3.2.2 As the SoC prototype board directly

As described in Section 3.3.2.1, one of the purposes of "MCU200T Evaluation Kit" is to be as the SoC board directly. To achieve this goal, there are several key points here:

- The "MCU200T Evaluation Kit" is supposed to be programmed with the Nuclei evaluation SoC (included RISC-V core).For the details of the SoC, please refer to document <Nuclei_Eval_SoC_Intro.pdf>, this document can be easily acquired in Nuclei User Center http://user.nucleisys.com.
- There is a separated SPI Nor Flash on the board, this SPI Nor Flash will be connected to the SoC's SPI interface, which support the XiP mode. The RISC-V Core in the FPGA will start to fetch the instructions (with XiP mode) from this Flash after reset.
- The SoC need two clock inputs, which are allocated at board as below:
  - The RTC clock is directly from the FPGA board 32.768KHz clock source.
  - The high-speed clock is from the PLL output (in FPGA project). The original clock source of PLL is from the FPGA board 100MHz clock source.
- The SoC have the "System Reset" pin, and the FPGA project mapped the pin to the board MCU_RESET button, as depicted in the Figure 3.7 mark [19].
- The SoC have the "PoR Reset" pin, and the FPGA project mapped the pin to the board FPGA_RESET button, as depicted in the Figure 3.7 mark [18].
- The SoC have 32 GPIO pins, and the FPGA project mapped these GPIO pins to the board exposed connectors.
- There are 3 LED lights on the board, they are wired to 3 GPIO pins respectively, Red light to GPIO 19, Green light to GPIO 21, and Blue light to GPIO22.
- The SoC have JTAG pins and UART pins (pin-mux to GPIO_16 and GPIO_17 of the SoC), and the FPGA project mapped these pins to the board MCU_JTAG interface socket, as depicted in the Figure 3.7 mark [4]. The "MCU200T Debugger Kit" can be connected to this socket, as depicted in Figure 3.8, and then be used to debug the RISC-V core (burned inside FPGA).

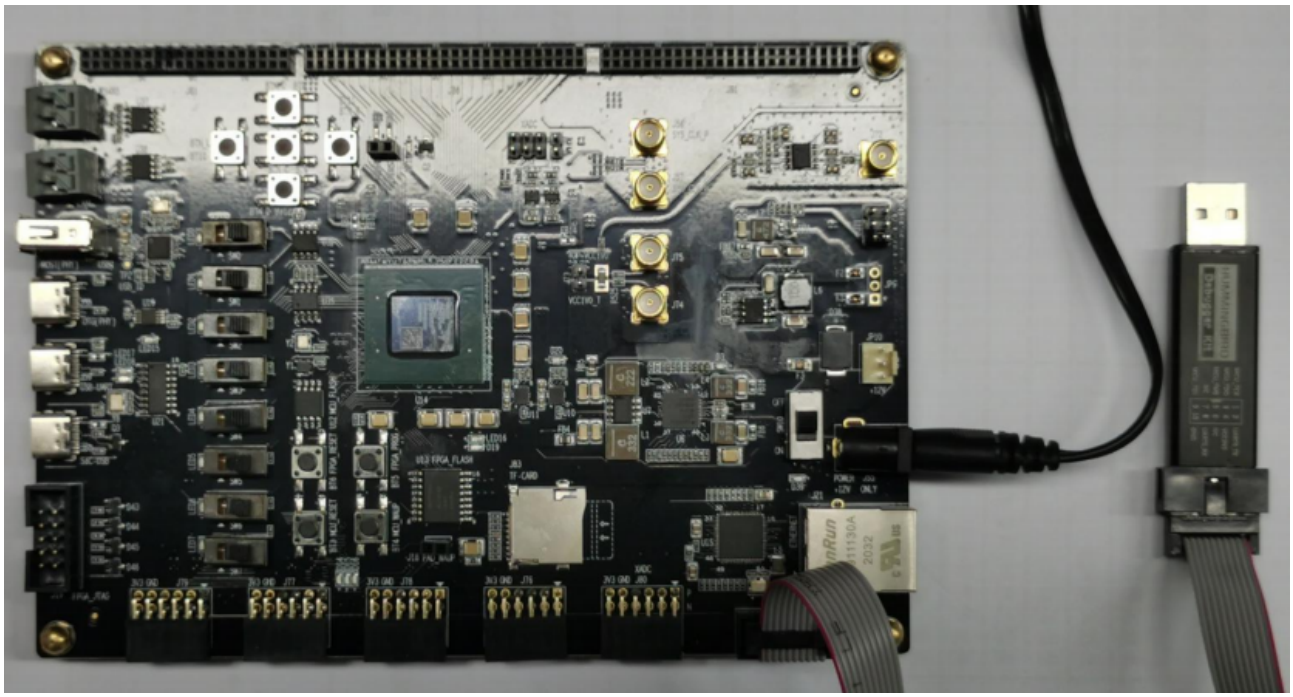

Fig. 4.8: Connecting MCU200T FPGA board and Debugger Kit with PC

4.3.2.3 As regular FPGA board resource

As described in Section 3.3.2.1, one of the purposes of "MCU200T Evaluation Kit" is to be as the regular FPGA board. To achieve this goal, there are push buttons , switches and LED lights, as depicted in the Figure 3-7 mark [17] [16] [15]. User can utilize these on-board resource to control the SoC, for examples as below:

- The switches (as depicted in the Figure 3-7 mark [16]) connect directly to FPGA GPIO. User can rewrite the constraint file(.xdc) to connect it to MCU_GPIOx pins, then to use the switch to control the MCU_GPIO input value.
- The LED lights (as depicted in the Figure 3-7 mark [15]) connect directly to FPGA GPIO.User can rewrite the constraint file(.xdc) to connect it to MCU_GPIOx pins, then to use the MCU_GPIO output value to control the LED lights.
- The push buttons (as depicted in the Figure 3-7 mark [17]) connect directly to FPGA GPIO. User can rewrite the constraint file(.xdc) to connect it to MCU_GPIOx pins, then to use the button to control the MCU_GPIO input value.

4.3.3 Generate the Bitstream file (MCS format)

The FPGA board has the SPI Flash to store FPGA's Bitstream (MCS format). If the users want to re-program the FPGA with the Bitstream file, Nuclei have provided the completed FPGA project (for each IP product), and then use Xilinx Vivado to generate the Bitstream file (MCS format).

For the detailed steps how to generate the Bitstream, please refer to each IP product's FPGA Prototype QuickStart document, which can be easily acquired in Nuclei User Center http://user.nucleisys.com.

4.3.4 Program the Bitstream (MCS format) into FPGA

After generating the Bitstream file (MCS format), then we can program it into the FPGA's SPI Flash.

Here are the steps how to program it with Vivado:

- Step1: Use the USB-JTAG Programming cables to connect the PC and the FPGA board's FPGA_JTAG connector, the FPGA_JTAG connector is the left yellow color highlighted box as depicted in the Figure 3-9.
- Step2: Use the USB cable to connect the Power source and the FPGA board's 12V Power interface,the Power interface is the right yellow color highlighted box as depicted in the Figure 3-9. And then manually turn on the power by switching it to ON status, the LED light along with it will be on after it.
- Step3: Open Xilinx Vivado, and open its Hardware Manager (as depicted in Figure 3.10), it will automatically recognize the board (via USB-JTAG Programming cables), as depicted in Figure 3.11.
- Step4: Right click the FPGA Device name, and choose the "Add Configuration Memory Device", as depicted in Figure 3.12.
- Step5: Select the Flash parameters, as depicted in Figure 3.13.
  - Part n25q128-3.3v
  - Manufacturer Micron
  - Family n25q
  - Type spi
  - Density 128
  - Width x1 x2 x4
- Step6: There will be a box pop out: "Do you want to program the configuration memory device now" choose "OK".
- Step7: Choose the "Configuration file" as depicted in Figure 3.14, and add your MCS files (e.g.,n205_rls_pkg/n205_cct/fpga/hbirdkit/system.mcs), and then select OK, it will start to program FPGA's SPI Flash, it will take a few seconds to wait.
- Step8: If the above step succeeded, user can push FPGA board's "FPGA_PROG" button (as depicted in the Figure 3.7 mark [20]), it will trigger the FPGA to load the Bitstream from the SPI Flash into the FPGA chip and make it start to wait.

- Step9: If the above steps succeeded, then user can pull out the USB-JTAG Programming cables from “FPGA JTAG” connector, unless user want to re-program FPGA again.

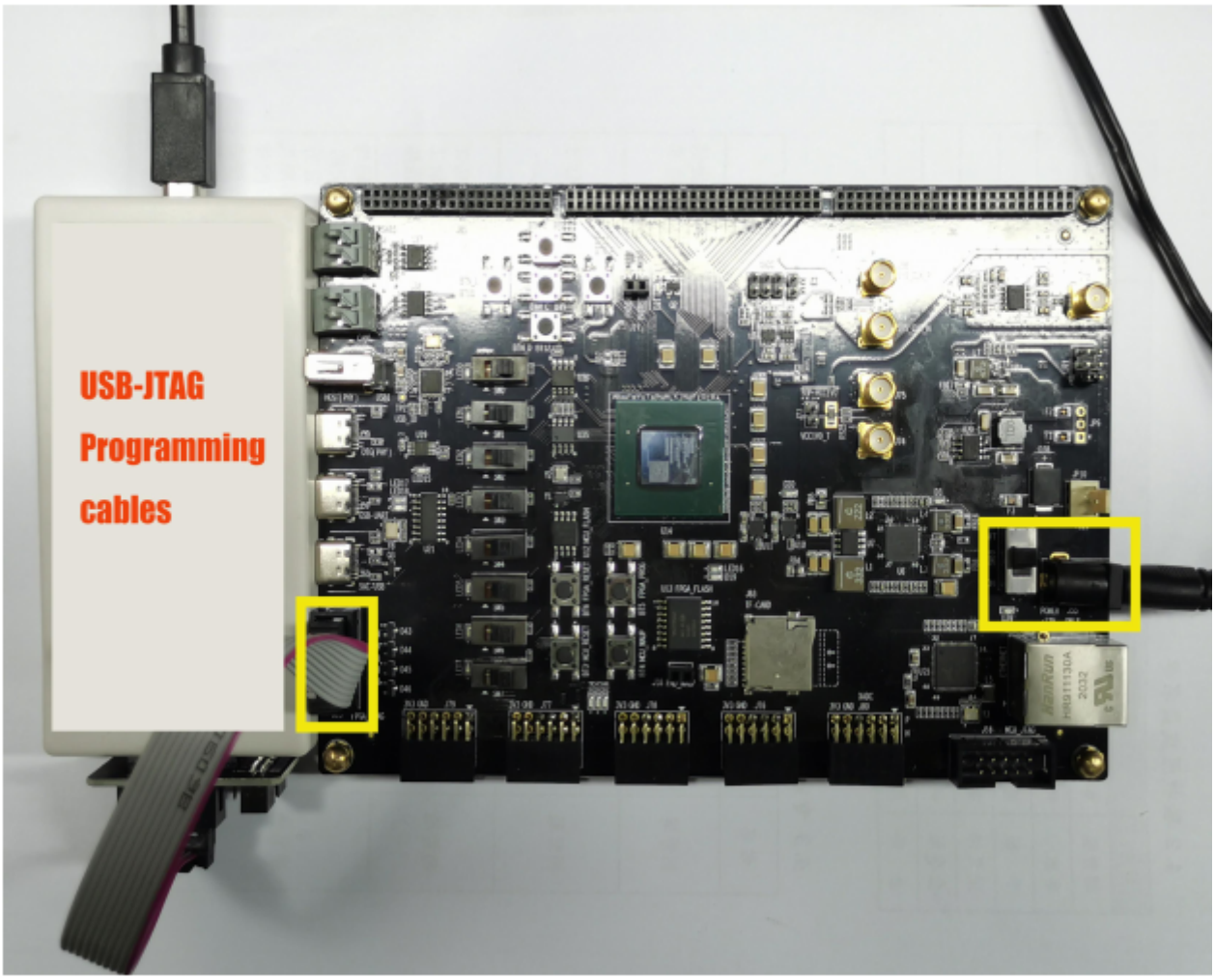

Fig. 4.9: The FPGA_JTAG connector and the 12V Power interface

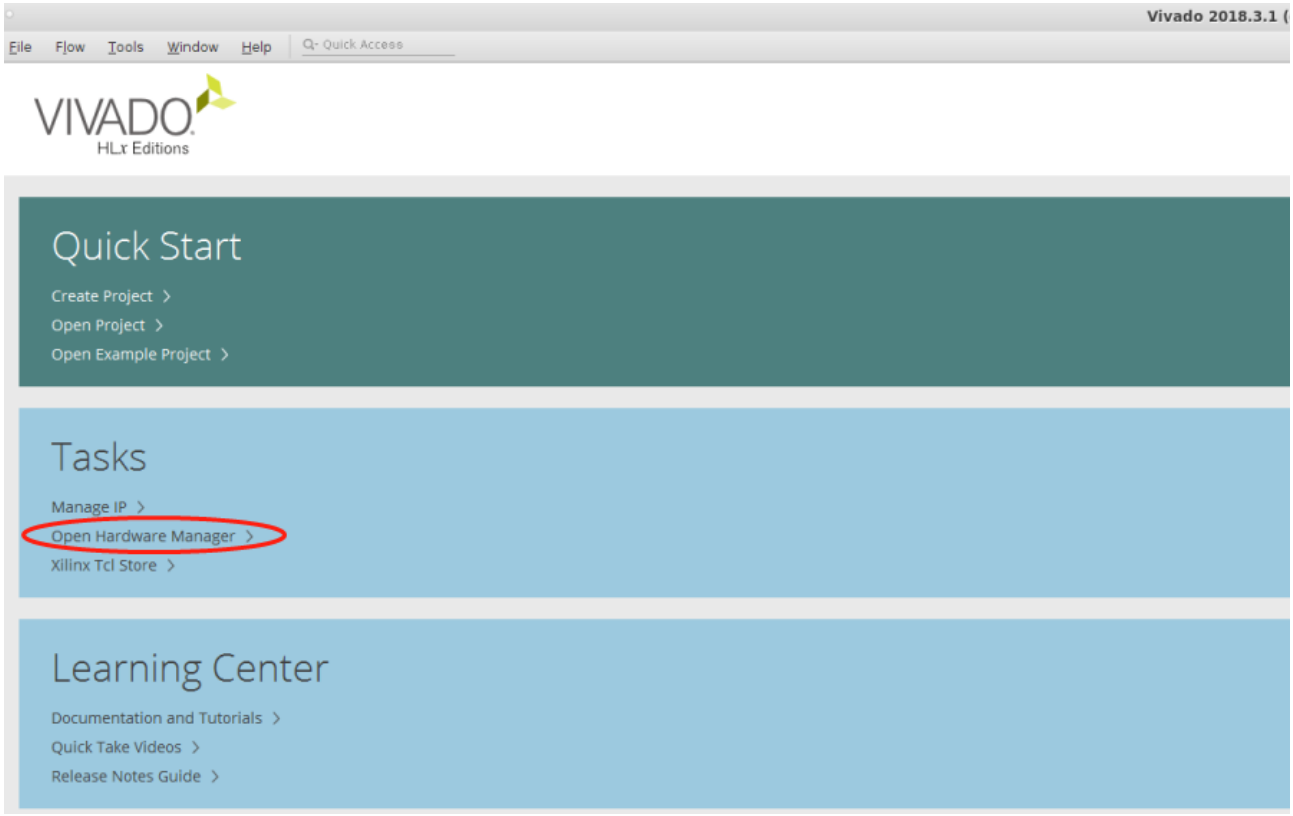

Fig. 4.10: The Hardware Manager of Vivado

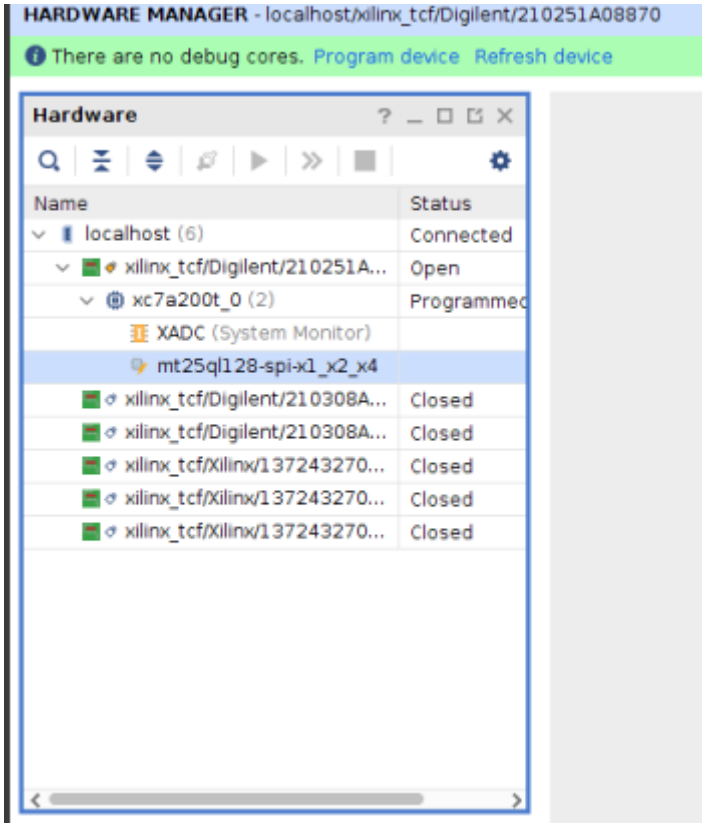

Fig. 4.11: Use Vivado Hardware Manager to connect FPGA

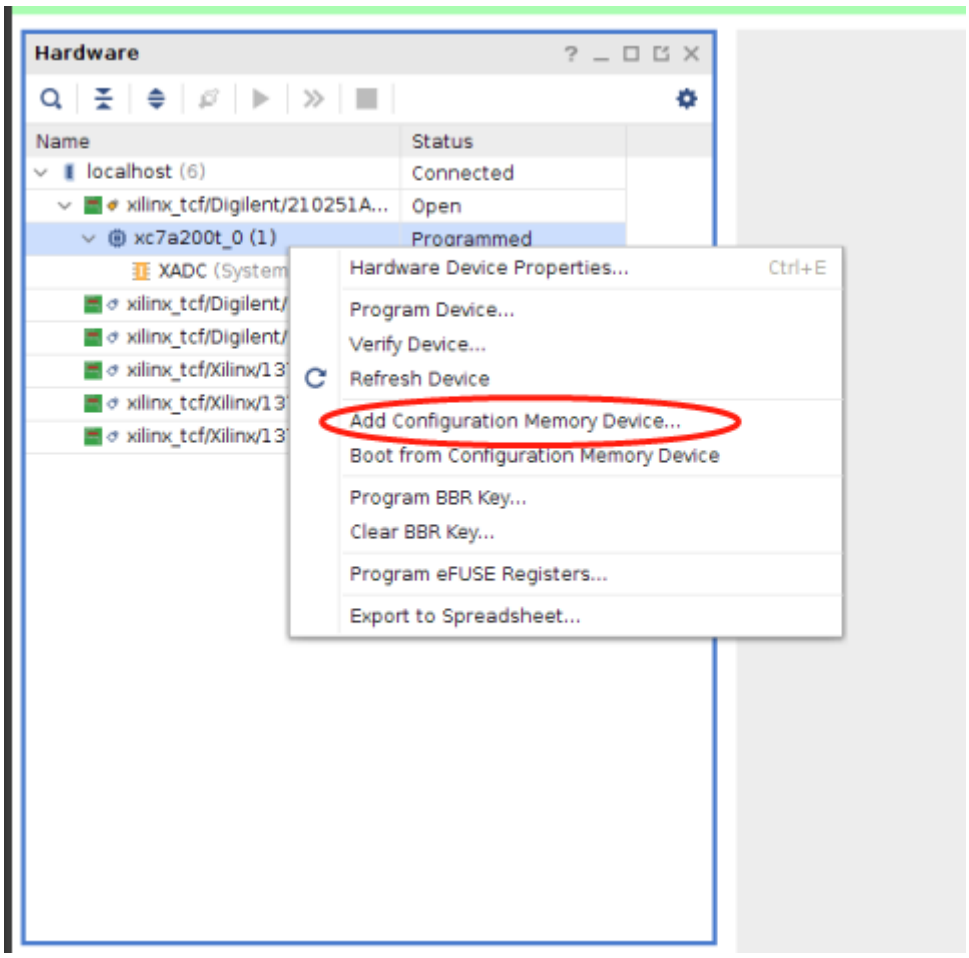

Fig. 4.12: Add Configuration Memory Device

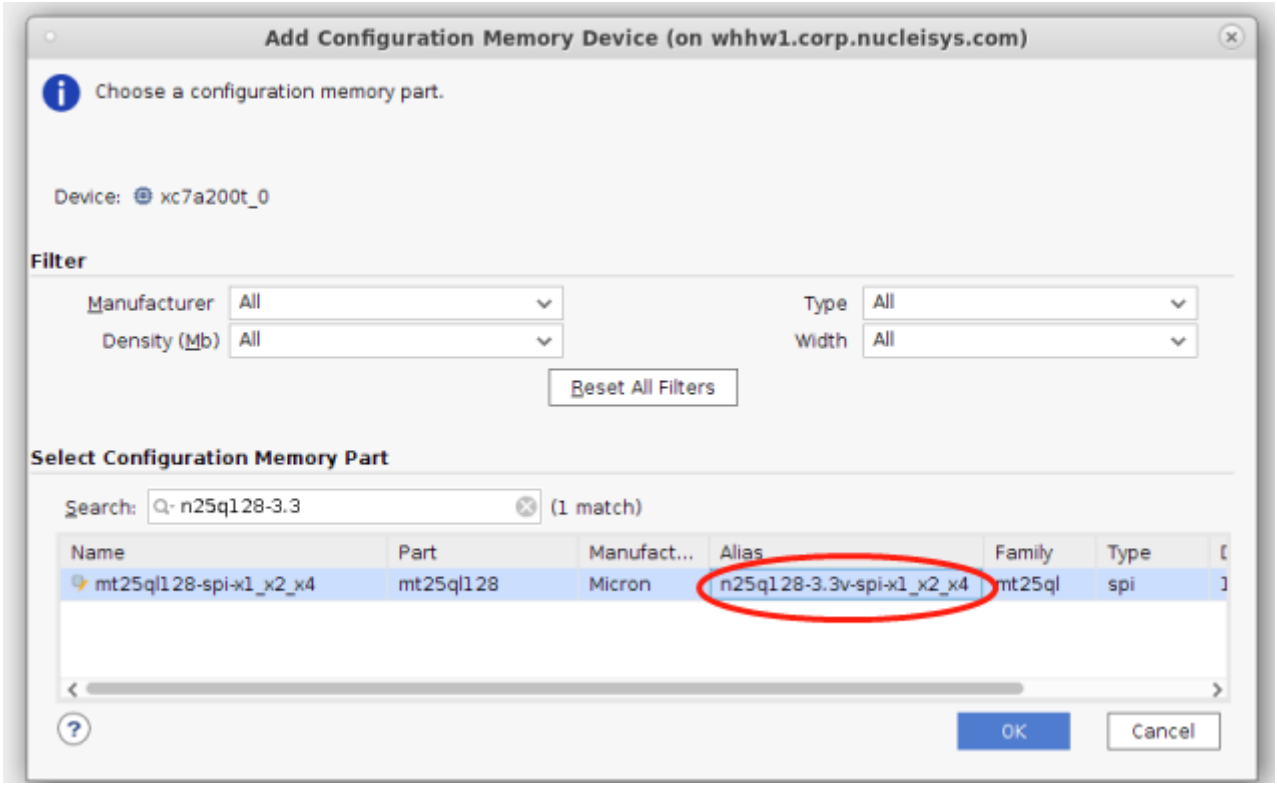

Fig. 4.13: Select the Flash Types

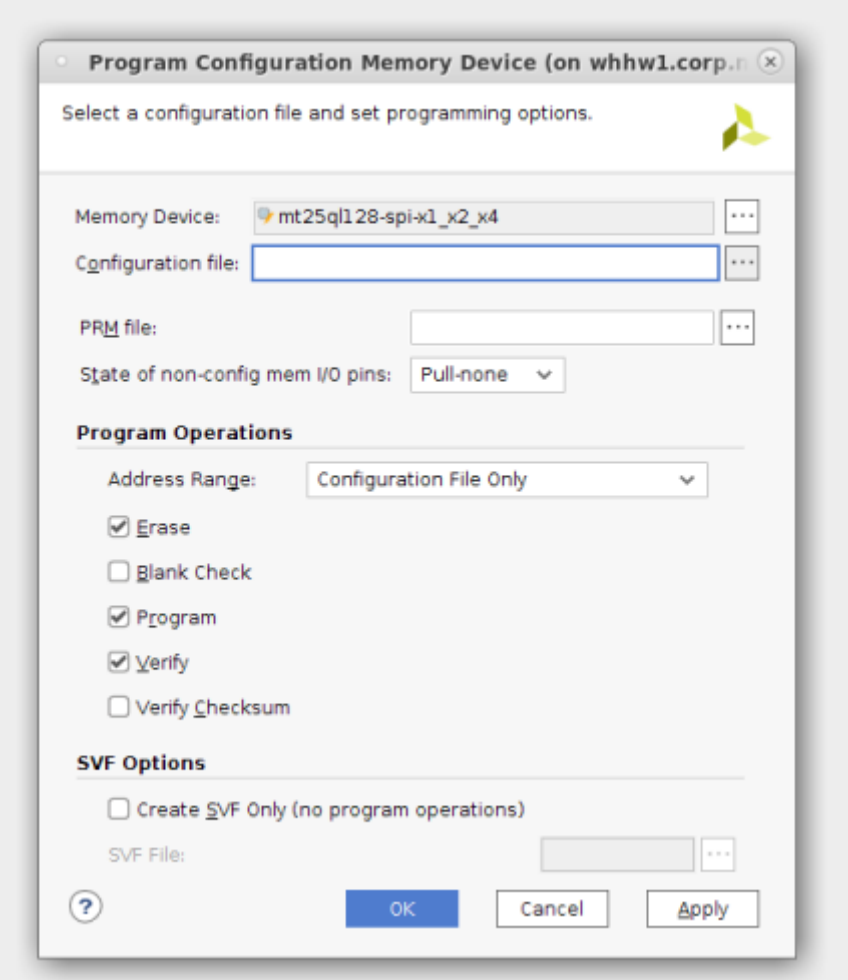

Fig. 4.14: Add the Configuration Files

4.4 DDR200T Evaluation FPGA Board

4.4.1 Overview of FPGA board

Nuclei have customized a FPGA evaluation board (called DDR200T Evaluation Kit), which is to make"One board serves two purposes":

- As the SoC prototype board directly: - If the FPGA have been pre-burned (programmed) with "Nuclei evaluation SoC", this board can be worked as a SoC prototype directly. Since the board has been designed with buttons and extended ports names in line with the SoC GPIO pin name, the embedded software engineers can directly use this board without knowing any FPGA hardware knowledge.
  - To easy the writing, the "Nuclei evaluation SoC" will be shorted as "the SoC" in this document thereby.
- As the regular FPGA board
- For those hardware engineers who know FPGA hardware knowledge, this board can be used as regular FPGA board, which can be used to burn (program) any Verilog HDL design.

4.4.2 Board Hardware Resources

4.4.2.1 Overall Introduction of Hardware Resources

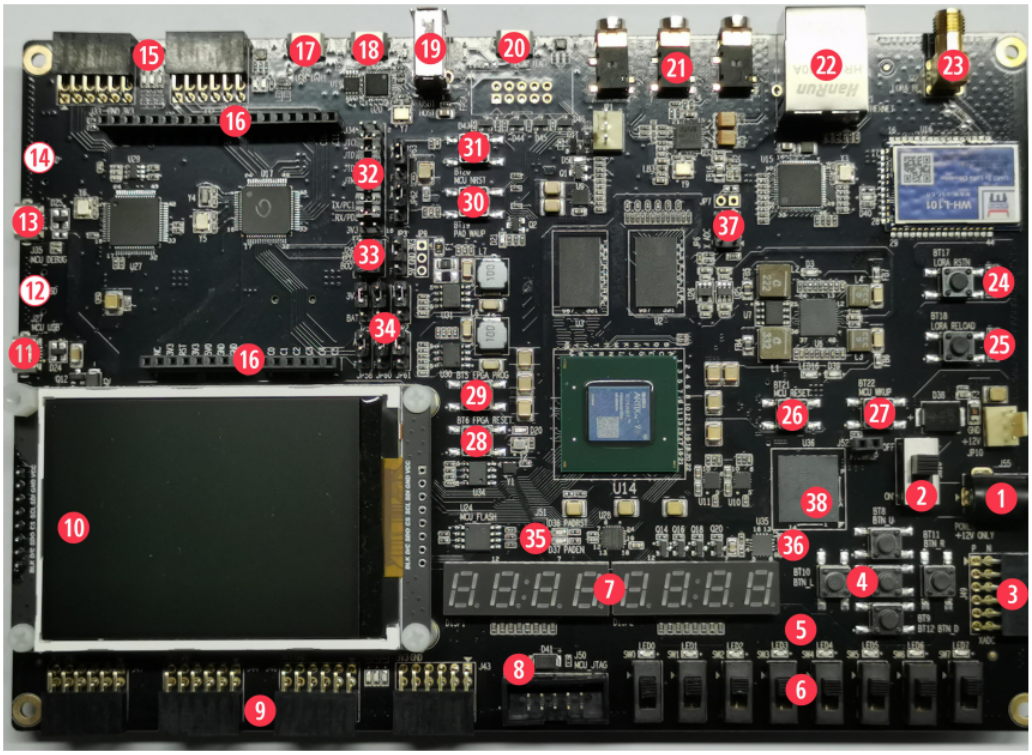

Fig. 4.15: Board Hardware Resources

DDR200T Evaluation Kit is an high-level FPGA board based on Xilinx FPGA, as depicted in Figure 3-15, the hardware resource (marked as with different numbers in above picture) of this board is described as follows.

The basic hardware resources for FPGA:

- Xilinx FPGA Chip (XC7A200T-2FBG484I).
- 4Gb DDR3L SDRAM with 32-bit bus (MT41K128M16JT-125K DDR III).
- Two Crystal Oscillators, Y1 is for 100MHz Clock, Y2 is for 32.768K RTC Clock.
- FPGA_FLASH, independent SPI Flash chip for FPGA to store its Bitstream (MCS format).
  - The Xilinx Vivado support the Bitstream to be burned into external SPI Flash with MCS format, the FPGA hardware will automatically re-load the Bitstream from external Flash into the FPGA each time after reset. By this way, FPGA can make sure its Bitstream will not be lost after power-down (since Flash will retain its Bitstream). Please refer to Section 3.4.4 for the steps how to burn MCS format Bitstream into external Flash.
- The FPGA_JTAG interface, which is the on-board USB JTAG programmer for programming the Bitstream of the FPGA. (Mark: [20])
- Separated DC 12V power supply and a power switch. (Mark:[1] & [2])
- The FPGA_PROG button to force the FPGA to re-load the Bitstream from the external Flash. (Mark:[29])

The pre-defined hardware resources for the RISC-V Core and the SoC are as below. Please refer to Section 3.4.2.2 for more details of how these resource work.

- The MCU_FLASH, independent Flash chip for RISC-V core (burned inside FPGA) to store its instruction programs or read-only data.
- The MCU_JTAG socket, which is the on-board JTAG interface socket for RISC-V Core debugging. (Mark:[8])
- The MCU_GPIO ports x4 (Compatible with PMOD interface), for 32 GPIO pins of the SoC. (Mark:[9])
- The FPGA_RESET button for PoR reset of the SoC. (Mark: [28])

- The MCU_RESET button for System reset of the SoC. (Mark: [26])

Other hardware resources as regular FPGA board resource:

- User LEDs x8. (Mark:[5])
- User Push Buttons x5 . (Mark:[4])
- User DIP Switch (8-position) . (Mark:[6])
- XADC port (Compatible with PMOD interface). (Mark:[3])
- Digit 7-Segment LED Display(8-position). (Mark:[7])
- 2.8inch LCD module . (Mark:[10])
- User IIC EEPROM: 24LC04
- FPGA SD Card Slot . (Mark:[12])
- UART To USB Bridge .(Mark:[17])
- FPGA USB-OTG port .(Mark:[18])
- FPGA USB-HOST port .(Mark:[19])
- FPGA Digital audio codec interface . (Mark:[21])
- 10/100/1000 Mbps Ethernet RJ45 port(GMII). (Mark:[22])
- LORA module RF SMA port . (Mark:[23])
- LORA module RESET Push Buttons . (Mark:[24])
- LORA module RELOAD Push Buttons . (Mark:[25])
- FPGA 8G EMMC. (Mark:[38]

RISC-V MCU (GD32VF103) resource:

- GD32VF103 USB port . (Mark:[11])
- GD32VF103 on-board USB JTAG Debugger port . (Mark:[13])
- GD32VF103 SD Card Slot . (Mark:[14])
- GD32VF103 PMOD port . (Mark:[15])
- GD32VF103 Arduino port . (Mark:[16])
- GD32VF103 MCU_NRST Push Buttons . (Mark:[38])
- GD32VF103 PA0_WKUP Push Buttons . (Mark:[31])
- GD32VF103 Debugger Select-jumper Header . (Mark:[32])
- GD32VF103 BOOT-jumper Header . (Mark:[33])
- GD32VF103 GPIO jumper Header(To FPGA GPIO or Arduino port) . (Mark:[34])

4.4.2.2 As the SoC prototype board directly

As described in Section 3.4.2.1, one of the purposes of "DDR200T Evaluation Kit" is to be as the SoC board directly. To achieve this goal, there are several key points here:

- The "DDR200T Evaluation Kit" is supposed to be programmed with the Nuclei evaluation SoC (included RISC-V core). For the details of the SoC, please refer to document <Nuclei_Eval_SoC_Intro.pdf>, this document can be easily acquired in Nuclei User Center http://user.nucleisys.com.
- There is a separated SPI Nor Flash on the board, this SPI Nor Flash will be connected to the SoC's SPI interface, which support the XiP mode. The RISC-V Core in the FPGA will start to fetch the instructions (with XiP mode) from this Flash after reset.
- The SoC need two clock inputs, which are allocated at board as below:
  - The RTC clock is directly from the FPGA board 32.768KHz clock source.

  - The high-speed clock is from the PLL output (in FPGA project). The original clock source of PLL is from the FPGA board 100MHz clock source.

- The SoC have the "System Reset" pin, and the FPGA project mapped the pin to the board MCU_RESET button, as depicted in the Figure 3.15 mark [26].
  - The SoC have the "PoR Reset" pin, and the FPGA project mapped the pin to the board FPGA_RESET button, as depicted in the Figure 3.15 mark [28]
  - The SoC have 32 GPIO pins, and the FPGA project mapped these GPIO pins to the board exposed PMOD connectors
  - There are 3 LED lights on the board, they are wired to 3 GPIO pins respectively, Red light to GPIO 19, Green light to GPIO 21, and Blue light to GPIO22.
  - The SoC have JTAG pins and UART0 pins (pin-mux to GPIO_16 and GPIO_17 of the SoC),and the FPGA project mapped these pins to the board MCU_JTAG interface socket, as depicted in the Figure 3.15 mark [8]. The "DDR200T Debugger Kit" can be connected to this socket, as depicted in Figure 3.16, and then be used to debug the RISC-V core (burned inside FPGA)

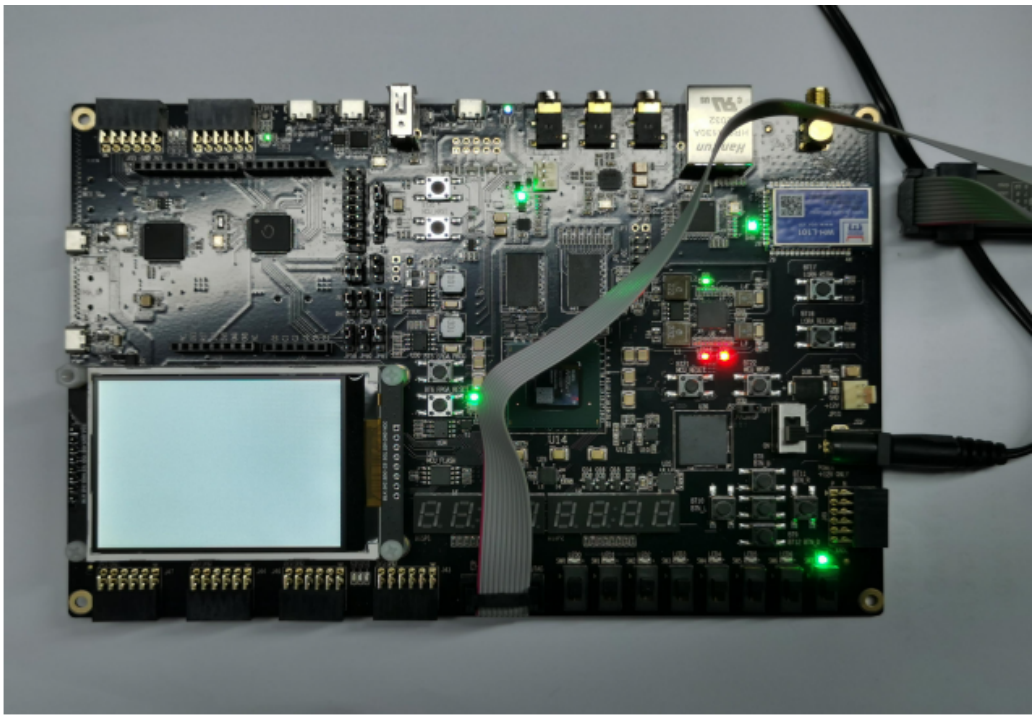

Fig. 4.16: Connecting DDR200T FPGA board and Debugger Kit with PC

4.4.2.3 As regular FPGA board resource

As described in Section 3.4.2.1, one of the purposes of "DDR200T Evaluation Kit" is to be as the regular FPGA board. To achieve this goal, there are push buttons , switches and LED lights, as depicted in the Figure 3.15 mark [4] [6] [5]. User can utilize these on-board resource to control the SoC, for examples as below:

- The switches (as depicted in the Figure 3.15 mark [6]) connect directly to FPGA GPIO. User can rewrite the constraint file(.xdc) to connect it to MCU_GPIOx pins, then to use the switch to control the MCU_GPIO input value.
- The LED lights (as depicted in the Figure 3.15 mark [5]) connect directly to FPGA GPIO.User can rewrite the constraint file(.xdc) to connect it to MCU_GPIOx pins, then to use the MCU_GPIO output value to control the LED lights
- The buttons (as depicted in the Figure 3.15 mark [4]) connect directly to FPGA GPIO. User can rewrite the constraint file(.xdc) to connect it to MCU_GPIOx pins, then to use the button to control the MCU_GPIO input value.

4.4.3 Generate the Bitstream file (MCS format)

The FPGA board has the SPI Flash to store FPGA's Bitstream (MCS format). If the users want to re-program the FPGA with the Bitstream file, Nuclei have provided the completed FPGA project (for each IP product), and then use Xilinx Vivado to generate the Bitstream file (MCS format). For the detailed steps how to generate the Bitstream, please refer to each IP product's FPGA Prototype QuickStart document, which can be easily acquired in Nuclei User Center http://user.nucleisys.com.

4.4.4 Program the Bitstream (MCS format) into FPGA

After generating the Bitstream file (MCS format), then we can program it into the FPGA's SPI Flash.

Here are the steps how to program it with Vivado :

- Step 1 :Use the USB cable to connect the PC and the FPGA board's FPGA_JTAG interface, the FPGA_JTAG interface is the left red color highlighted box as depicted in the Figure 3.17.
- Step 2 :Use the USB cable to connect the Power source and the FPGA board's 12V Power interface, the Power interface is the right red color highlighted box as depicted in the Figure 3.17. And then manually turn on the power by switching it to ON status, the LED light along with it will be on after it.
- Step 3 :Open Xilinx Vivado, and open its Hardware Manager (as depicted in Figure 3-18), it will automatically recognize the board (via USB interface), as depicted in Figure 3.19.
- Step 4 :Right click the FPGA Device name, and choose the "Add Configuration Memory Device", as depicted in Figure 3.20.
- Step 5 :Select the Flash parameters, as depicted in Figure 3.21.
  - Part n25q128-3.3.v
  - Manufacturer Micron
  - Family n25q
  - Type spi
  - Density 128
  - Width x1 x2 x4
- Step 6 :There will be a box pop out: "Do you want to program the configuration memory device now?" choose "OK"
- Step 7 : Choose the "Configuration file" as depicted in Figure 3.22, and add your MCS files (e.g., n205_rls_pkg/n205_cct/fpga/hbirdkit/system.mcs), and then select OK, it will start to program FPGA's SPI Flash, it will take a few seconds to wait.
- Step 8 : If the above step succeeded, user can push FPGA board's "FPGA_PROG" button (as depicted in the Figure 3.15 mark [29]), it will trigger the FPGA to load the Bitstream from the SPI Flash into the FPGA chip and make it start to work.
- Step 9 : If the above steps succeeded, then user can pull out the USB cable from "FPGA JTAG" interface, unless user want to re-program FPGA again.

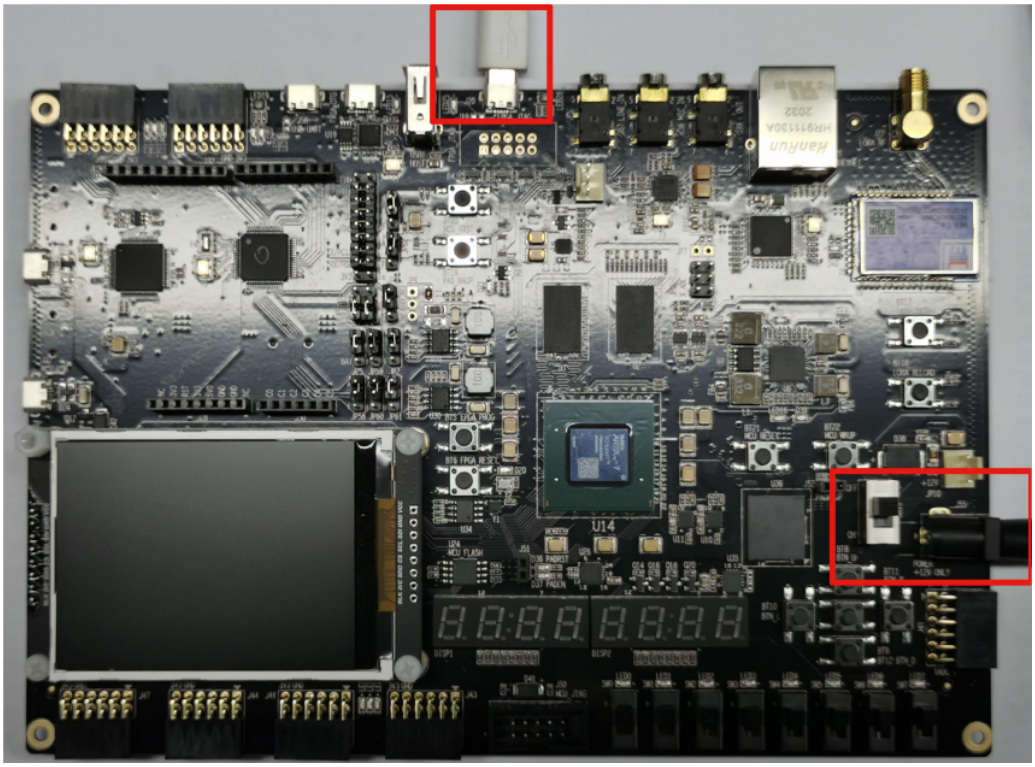

Fig. 4.17: The FPGA_JTAG interface and the 12V Power interface

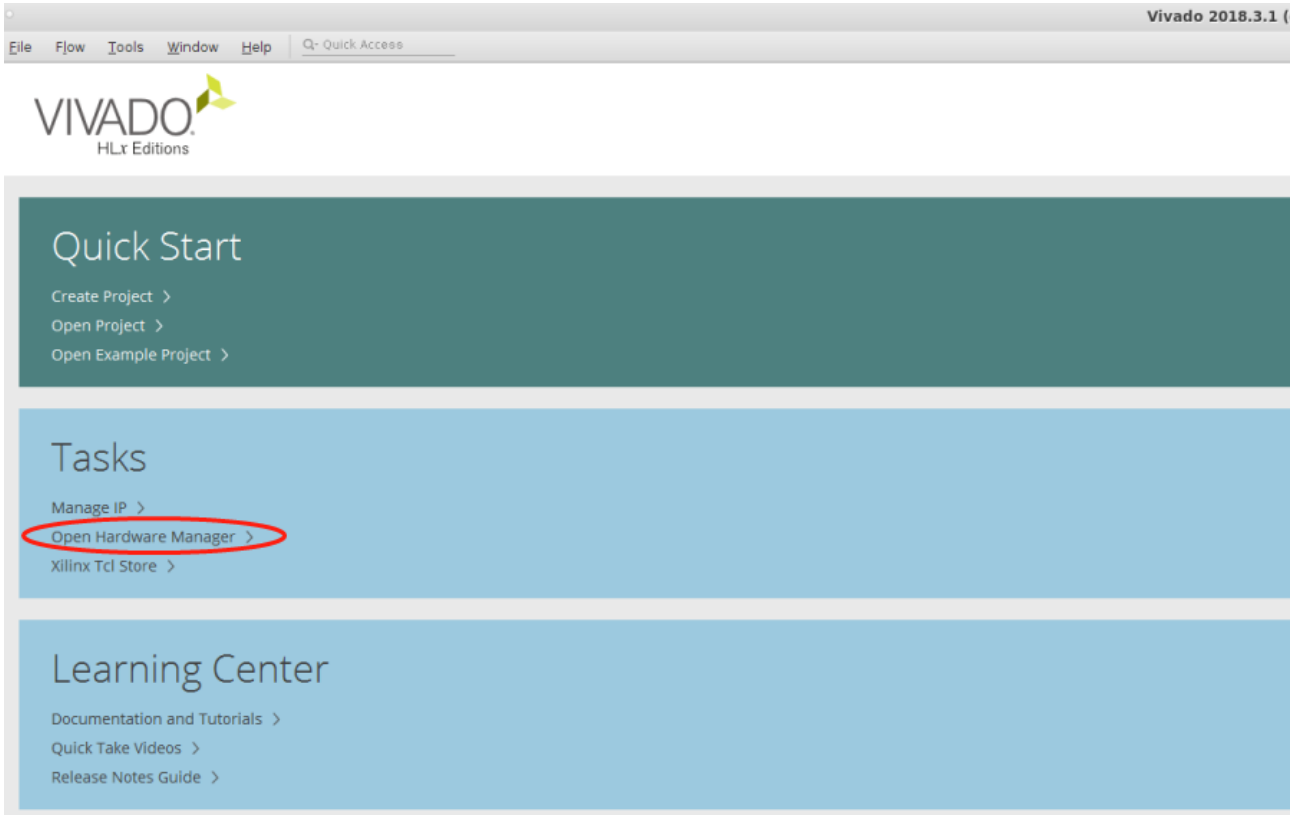

Fig. 4.18: The Hardware Manager of Vivado

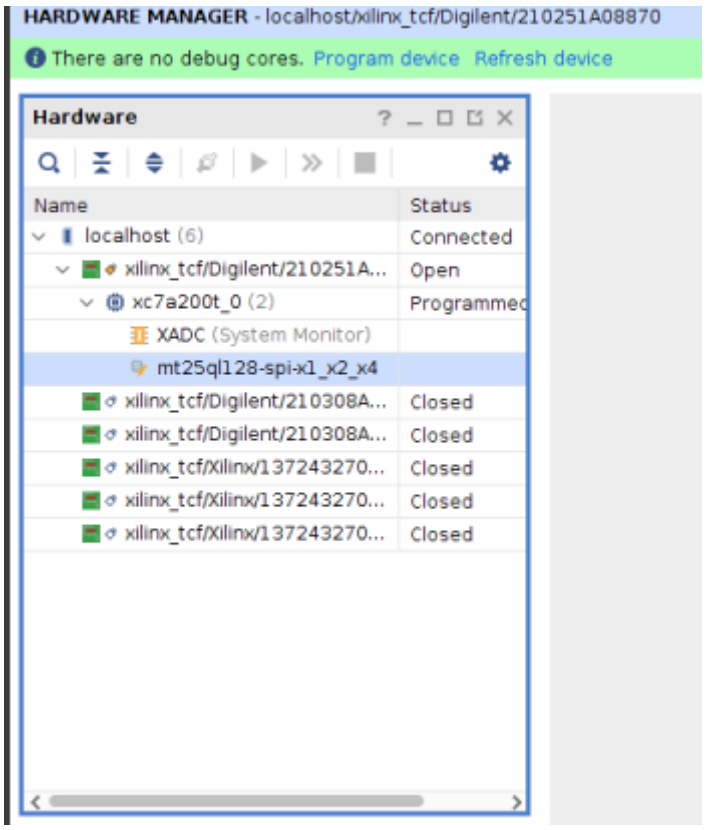

Fig. 4.19: Use Vivado Hardware Manager to connect FPGA

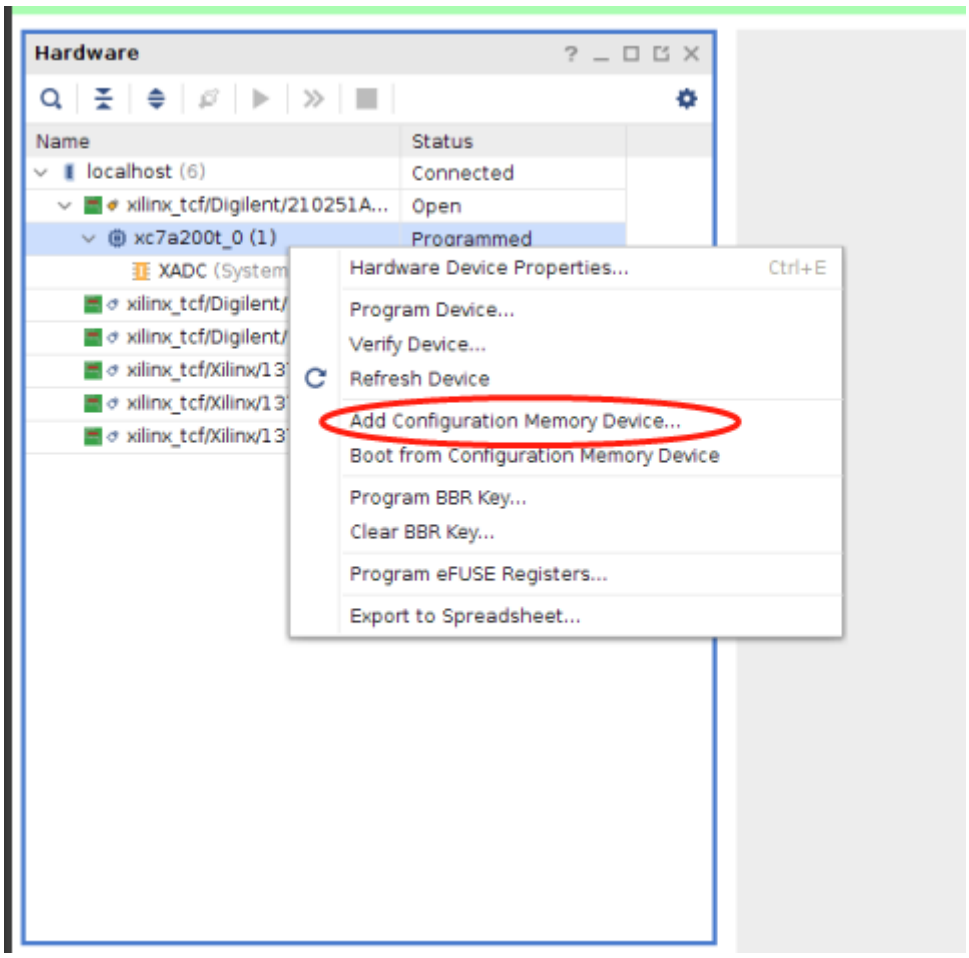

Fig. 4.20: Add Configuration Memory Device

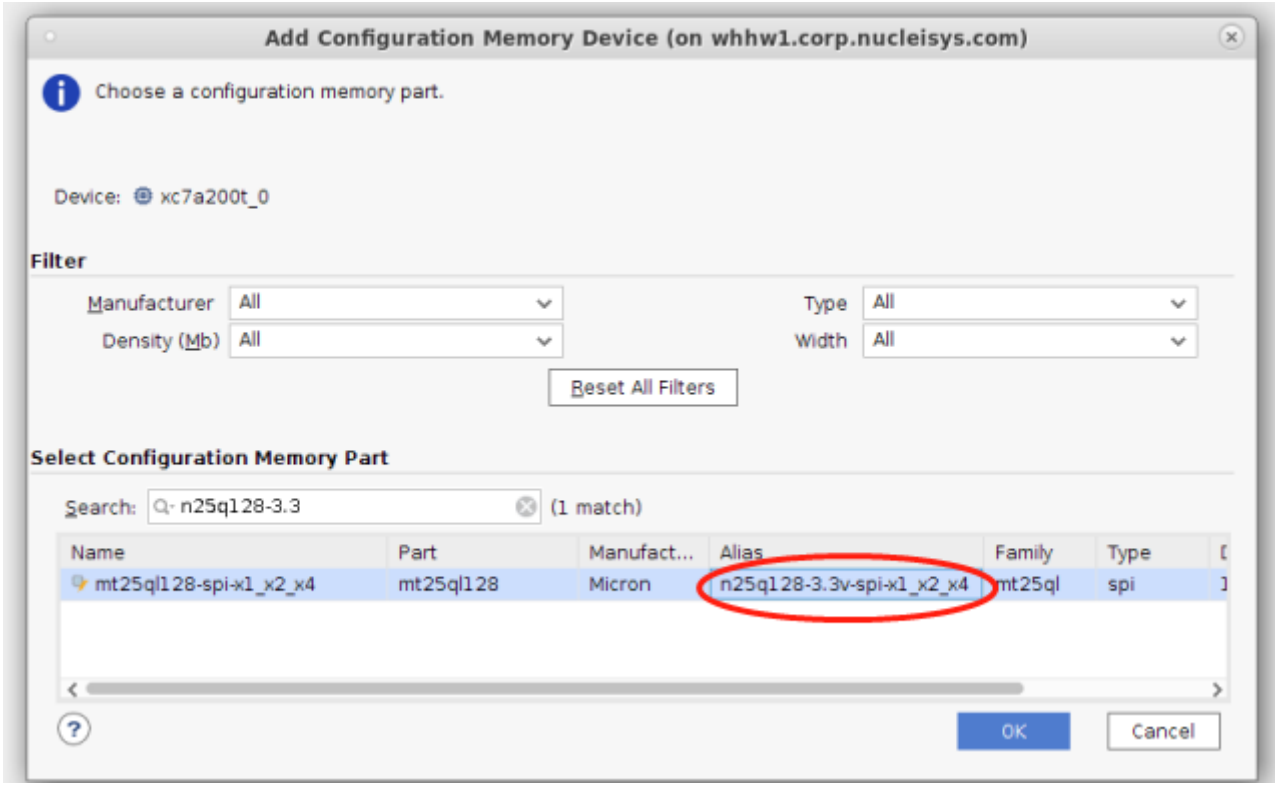

Fig. 4.21: Select the Flash Types

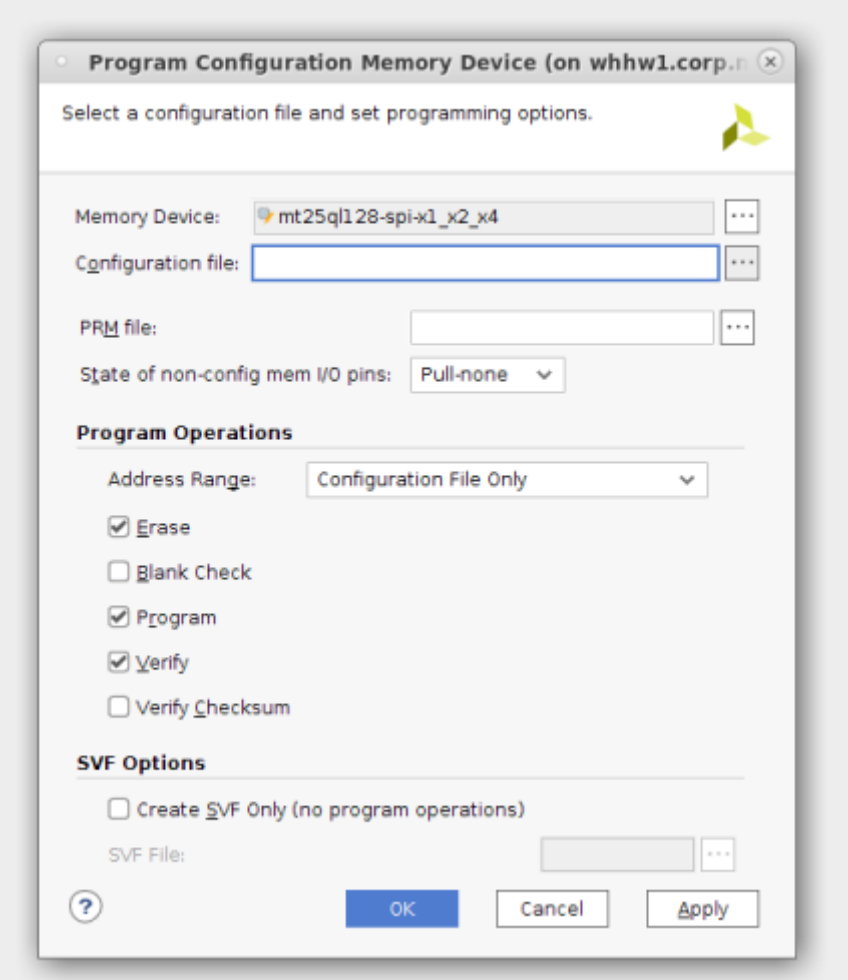

Fig. 4.22: Add the Configuration Files

4.5 KU060 Evaluation FPGA Board

4.5.1 Overview of FPGA board

Nuclei have customized a FPGA evaluation board (called KU060 Evaluation Kit), which is to make "One board serves two purposes":

- As the SoC prototype board directly:
  - If the FPGA have been pre-burned (programmed) with "Nuclei evaluation SoC" , this board can be worked as a SoC prototype directly. Since the board has been designed with buttons and extended ports names in line with the SoC GPIO pin name, the embedded software engineers can directly use this board without knowing any FPGA hardware knowledge.
  - To easy the writing,the "Nuclei evaluation SoC" will be shorted as "the SoC" in this document thereby.
- As the regular FPGA board.
  - For those hardware engineers who know FPGA hardware knowledge, this board can be used as regular FPGA board, which can be used to burn (program) any Verilog HDL design.

4.5.2 Board Hardware Resources

4.5.2.1 Overall Introduction of Hardware Resources

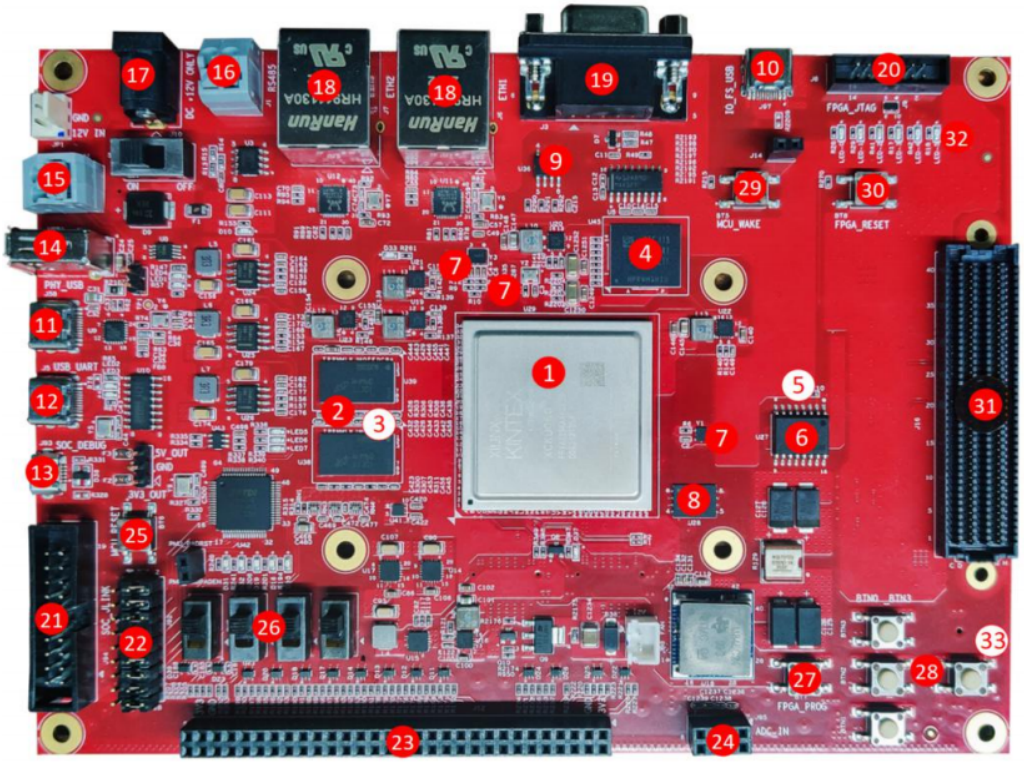

Fig. 4.23: Board Hardware Resources

KU060 Evaluation Kit is an high-level FPGA board based on Xilinx FPGA, as depicted in Figure 3-23, the hardware resource (marked as with different numbers in above picture) of this board is described as follows.

The basic hardware resources for FPGA:

- Xilinx FPGA Chip (XCKU060-FFVA1156-1C). (Mark: [1])
- 2GB DDR4 SDRAM with 64-bit bus (MT40A512M16LY-062E DDR4). (Mark: [2]& [3])
- 8GB eMMC. (Mark:[4])
- IPS6404L-SQ SPI/QSPI interface PSRAM. (Mark:[5])
- The MCU_FLASH, independent Flash chip for RISC-V core (burned inside FPGA) to store its instruction programs or read-only data. (Mark:[6])
- Two Crystal Oscillators, Y1 is for 100MHz Clock, Y2 is for 32.768K RTC Clock.(Mark:[7])
- FPGA_FLASH, independent SPI Flash chip for FPGA to store its Bitstream (MCS format). (Mark: [8])
  - The Xilinx Vivado support the Bitstream to be burned into external SPI Flash with MCS format, the FPGA hardware will automatically re-load the Bitstream from external Flash into the FPGA each time after reset. By this way, FPGA can make sure its Bitstream will not be lost after power-down (since Flash will retain its Bitstream). Please refer to Section 3.5.4 for the steps how to burn MCS format Bitstream into external Flash.
- User IIC EEPROM: 24LC04B. (Mark: [9])
- IO_FS_USB interface. (Mark: [10])
- USB2.0 interface (USB3315 PHY). (Mark: [11]&[14])
- USB to UART Interface (CH340). (Mark: [12])
- On-board HummingBird SoC Debugger(SOC_JTAG Select-jumper Header: default). (Mark:[13])
- CAN interface. (Mark:[15])

- RS485 interface. (Mark:[16])
- Separated DC 12V power supply and a power switch. (Mark:[17])
- Two 10/100/1000 Mbps Ethernet RJ45 port(RGMII). (Mark:[18])
- UART DB9 interface(Female). (Mark:[19])
- The FPGA_JTAG interface, which is the USB JTAG programmer Header for programming the Bitstream of the FPGA. (Mark: [20])
- SOC JLINK debug interface(SOC_JTAG Select-jumper Header). (Mark: [21])
- SOC_JTAG Select-jumper Header. (Mark: [22])
- FPGA GPIO, contains 51 GPIO ports (supporting 3.3V level standard). (Mark: [23])
- The 4-channel ADC analog input interface. (Mark: [24])
- The MCU_RESET button for System reset of the SoC. (Mark: [25])
- User DIP Switch (4-position) . (Mark:[26])
- The FPGA_PROG button to force the FPGA to re-load the Bitstream from the external Flash. (Mark:[27])
- User Push Buttons x4 . (Mark:[28])
- The MCU_WKUP button for System wake up of the SoC. (Mark: [29])
- The FPGA_RESET button for PoR reset of the SoC. (Mark: [30])
- LPC FMC interface (only support HP BANK level standard). (Mark: [31])
- User LEDs x6. (Mark:[32])
- FPGA SD Card Slot . (Mark:[33])

4.5.2.2 As the SoC prototype board directly

As described in Section 3.5.2.1, one of the purposes of "KU060 Evaluation Kit" is to be as the SoC board directly. To achieve this goal, there are several key points here:

- The "KU060 Evaluation Kit" is supposed to be programmed with the Nuclei evaluation SoC (included RISC-V core). For the details of the SoC, please refer to document <Nuclei_Eval_SoC_Intro.pdf>, this document can be easily acquired in Nuclei User Center http://user.nucleisys.com
- There is a separated SPI Nor Flash on the board, this SPI Nor Flash will be connected to the SoC's SPI interface, which support the XiP mode. The RISC-V Core in the FPGA will start to fetch the instructions (with XiP mode) from this Flash after reset.
- The SoC need two clock inputs, which are allocated at board as below:
  - The RTC clock is directly from the FPGA board 32.768KHz clock source.
  - The high-speed clock is from the PLL output (in FPGA project). The original clock source of PLL is from the FPGA board 100MHz clock source.
- The SoC have the "System Reset" pin, and the FPGA project mapped the pin to the board MCU_RESET button, as depicted in the Figure 3.23 mark [25].
- The SoC have the "PoR Reset" pin, and the FPGA project mapped the pin to the board FPGA_RESET button, as depicted in the Figure 3.23 mark [30].
- The SoC have 32 GPIO pins, and the FPGA project mapped these GPIO pins to the board exposed mark [23] connectors.
- There are 3 LED lights on the board, they are wired to 3 GPIO pins respectively, LED_0 to GPIO 19,LED_1 to GPIO 21, and LED_2 to GPIO22.
- The SoC have 4 pins JTAG and UART0 pins (pin-mux to GPIO_16 and GPIO_17 of the SoC), and the FPGA project mapped these pins to the On-board HummingBird SoC Debugger interface, as depicted in the Figure 3.23 mark [13]. The "KU060 Debugger Kit" can be connected to this socket, as depicted in Figure 3.24, and then be used to debug the RISC-V core (burned inside FPGA).

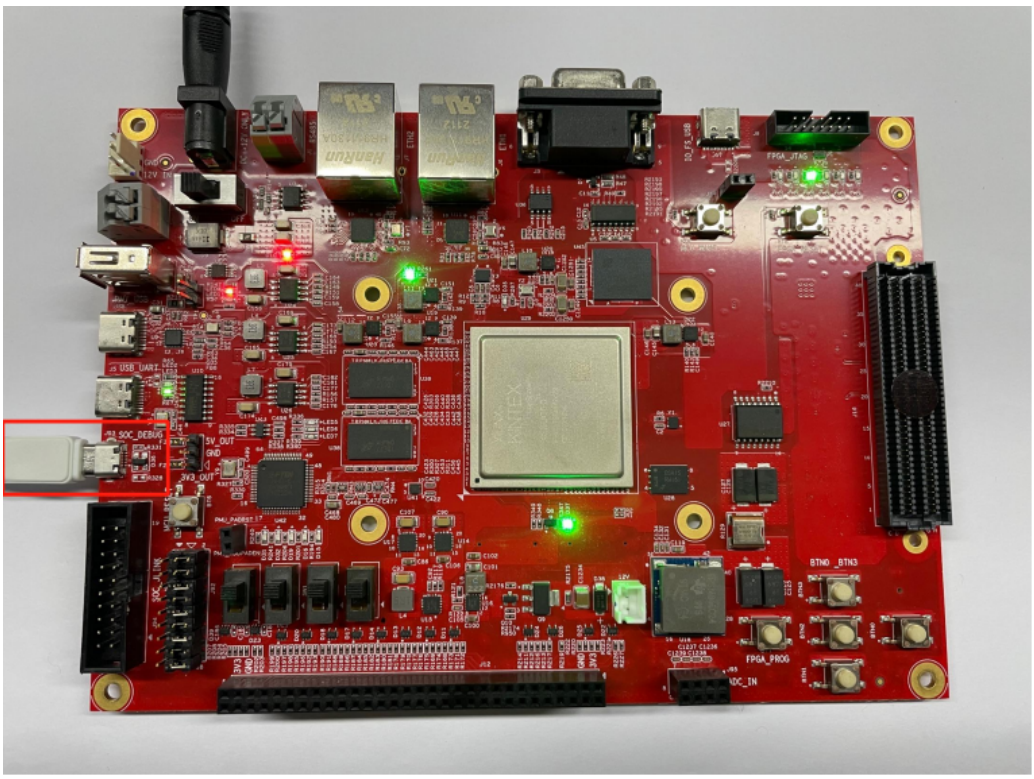

Fig. 4.24: The KU060 FPGA board's On-board HummingBird SoC Debugger interface

4.5.2.3 As regular FPGA board resource

As described in Section 3.5.2.1, one of the purposes of "KU060 Evaluation Kit" is to be as the regular FPGA board. To achieve this goal, there are push buttons , switches and LED lights, as depicted in the Figure 3.23 mark [28] [26] [32]. User can utilize these on-board resource to control the SoC, for examples as below:

- The switches (as depicted in the Figure 3.23 mark [26]) connect directly to FPGA GPIO. User can rewrite the constraint file(.xdc) to connect it to MCU_GPIOx pins, then to use the switch to control the MCU_GPIO input value.
- The LED lights (as depicted in the Figure 3.23 mark [32]) connect directly to FPGA GPIO.User can rewrite the constraint file(.xdc) to connect it to MCU_GPIOx pins, then to use the MCU_GPIO output value to control the LED lights.
- The buttons (as depicted in the Figure 3.23 mark [28]) connect directly to FPGA GPIO. User can rewrite the constraint file(.xdc) to connect it to MCU_GPIOx pins, then to use the button to control the MCU_GPIO input value.

4.5.3 Generate the Bitstream file (MCS format)

The FPGA board has the SPI Flash to store FPGA's Bitstream (MCS format). If the users want to re-program the FPGA with the Bitstream file, Nuclei have provided the completed FPGA project (for each IP product), and then use Xilinx Vivado to generate the Bitstream file (MCS format).

For the detailed steps how to generate the Bitstream, please refer to each IP product's FPGA Prototype QuickStart document, which can be easily acquired in Nuclei User Center http://user.nucleisys.com.

4.5.4 Program the Bitstream (MCS format) into FPGA

After generating the Bitstream file (MCS format), then we can program it into the FPGA's SPI Flash.

Here are the steps how to program it with Vivado:

- Step 1: Use the USB-JTAG Programming cables to connect the PC and the FPGA board's FPGA_JTAG interface, the FPGA_JTAG interface is the right red color highlighted box as depicted in the Figure 3.25.
- Step 2: Use the USB cable to connect the Power source and the FPGA board's 12V Power interface, the Power interface is the right red color highlighted box as depicted in the Figure 3.25. And then manually turn on the power by switching it to ON status, the LED light along with it will be on after it.
- Step 3: Open Xilinx Vivado, and open its Hardware Manager (as depicted in Figure 3.26), it will automatically recognize the board (via USB interface), as depicted in Figure 3.27.
- Step 4: Right click the FPGA Device name, and choose the "Add Configuration Memory Device", as depicted in Figure 3.28.
- **Step 5: Select the Flash parameters, as depicted in Figure 3.29.** Part n25q128-3.3v

  Manufacturer Micron

  Family n25q

  Type spi

  Density 128

  Width x1 x2 x4
- Step 6: There will be a box pop out: "Do you want to program the configuration memory device now?" choose "OK"
- Step 7: Choose the "Configuration file" as depicted in Figure 3.30, and add your MCS files (e.g.,n205_rls_pkg/n205_cct/fpga/hbirdkit/system.mcs), and then select OK, it will start to program FPGA's SPI Flash, it will take a few seconds to wait.
- Step 8: If the above step succeeded, user can push FPGA board's "FPGA_PROG" button (as depicted in the Figure 3.23 mark [27]), it will trigger the FPGA to load the Bitstream from the SPI Flash into the FPGA chip and make it start to work.
- Step 9: If the above steps succeeded, then user can pull out the USB-JTAG Programming cables from "FPGA JTAG" interface, unless user want to re-program FPGA again.

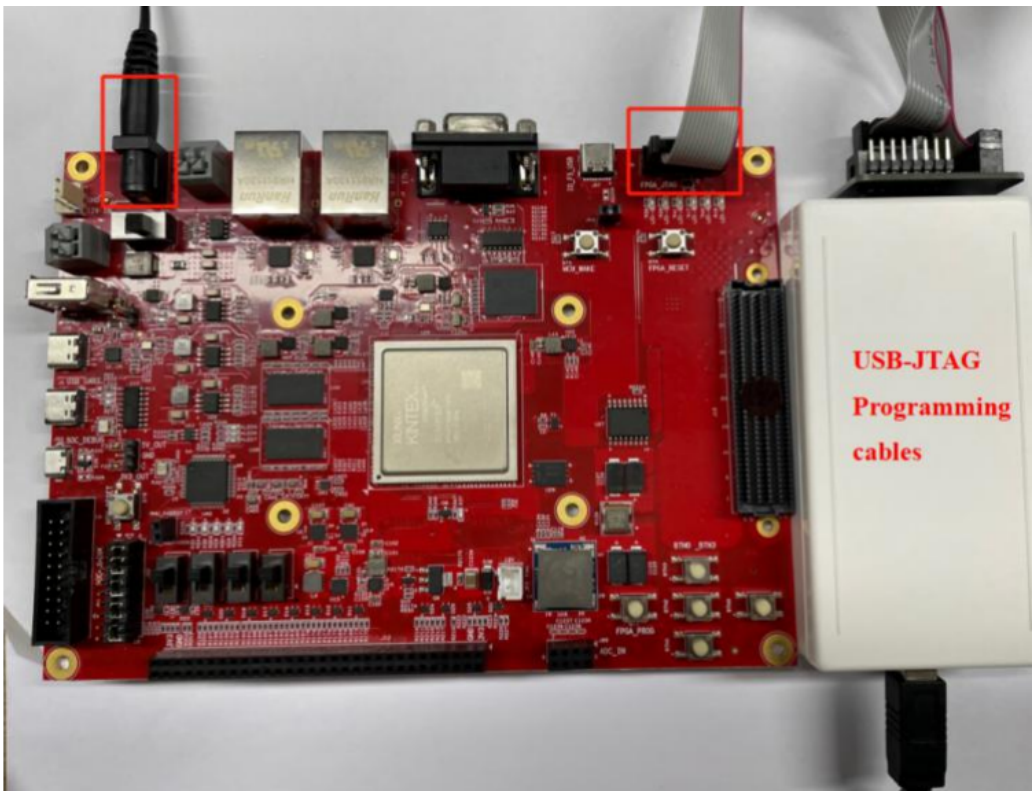

Fig. 4.25: The FPGA_JTAG connector and the 12V Power interface

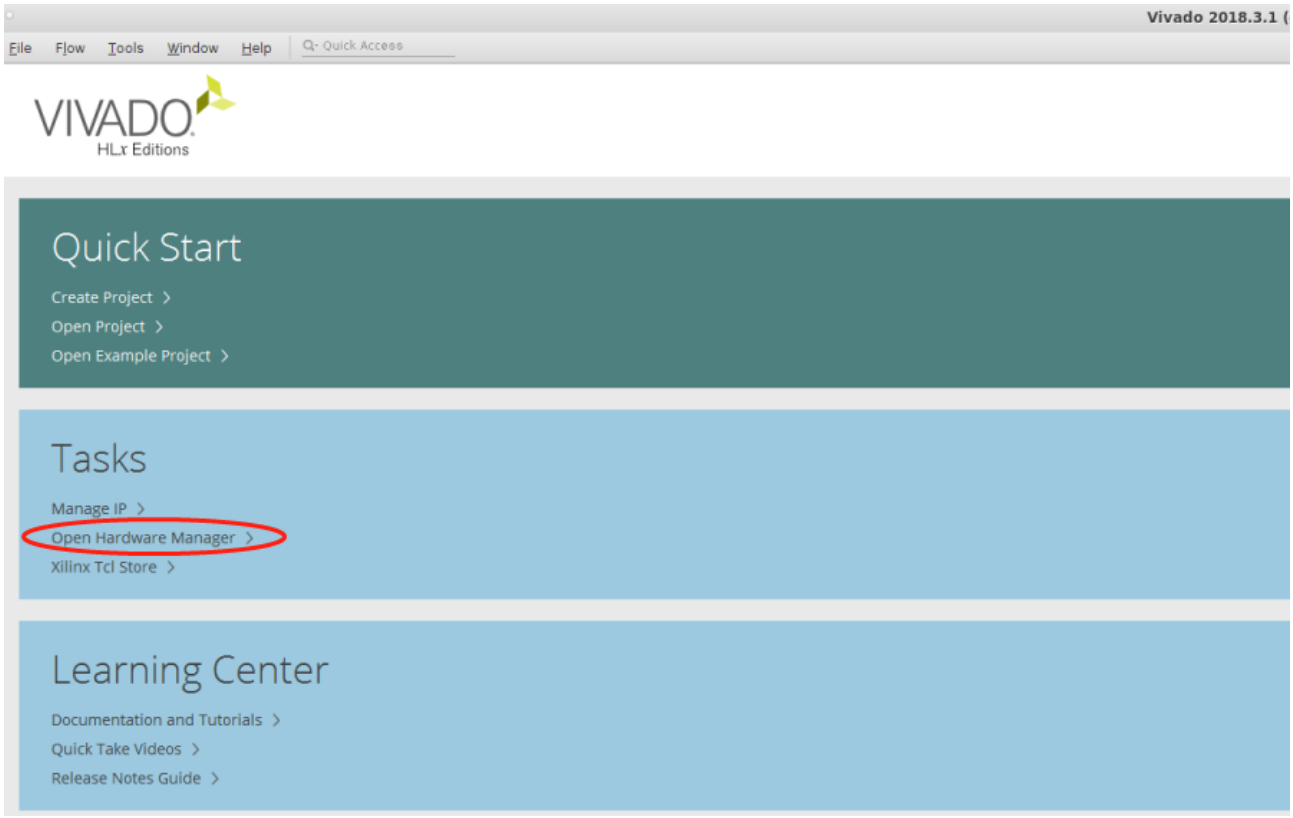

Fig. 4.26: The Hardware Manager of Vivado

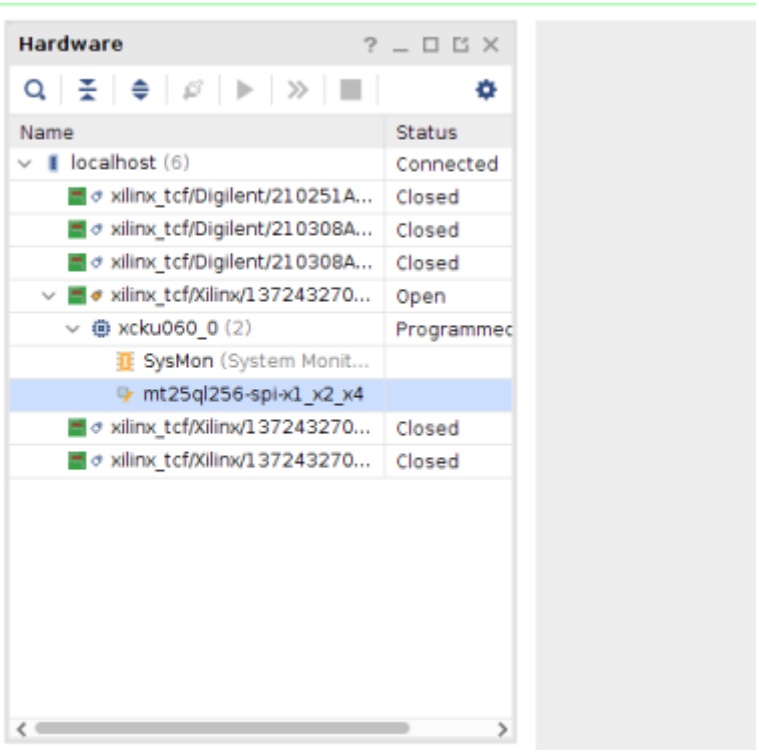

Fig. 4.27: Use Vivado Hardware Manager to connect FPGA

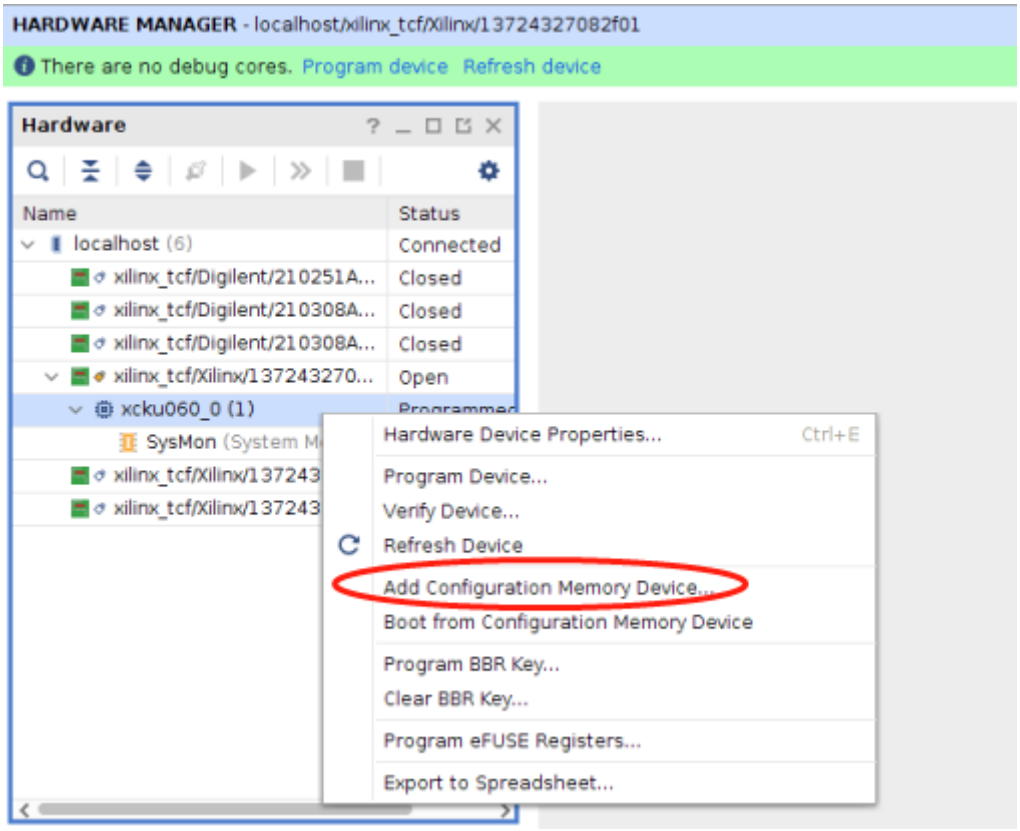

Fig. 4.28: Add Configuration Memory Device

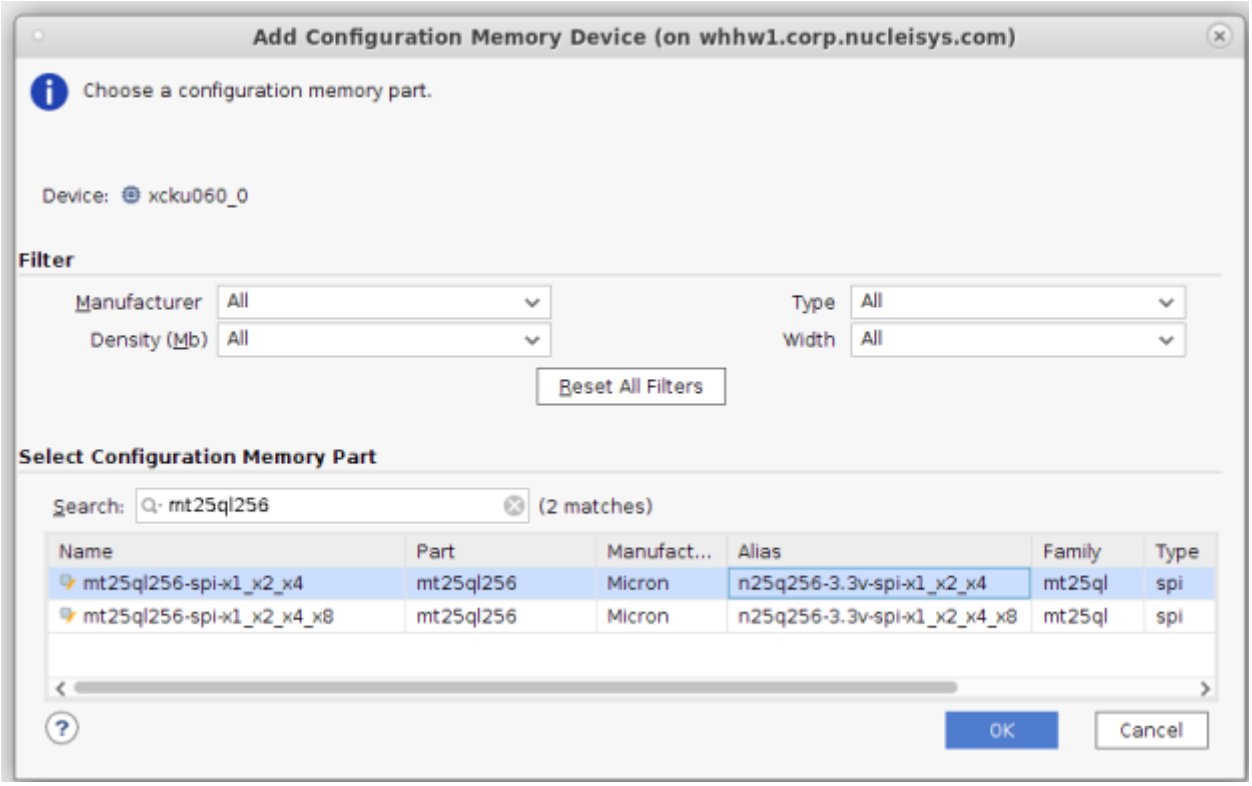

Fig. 4.29: Select the Flash Types

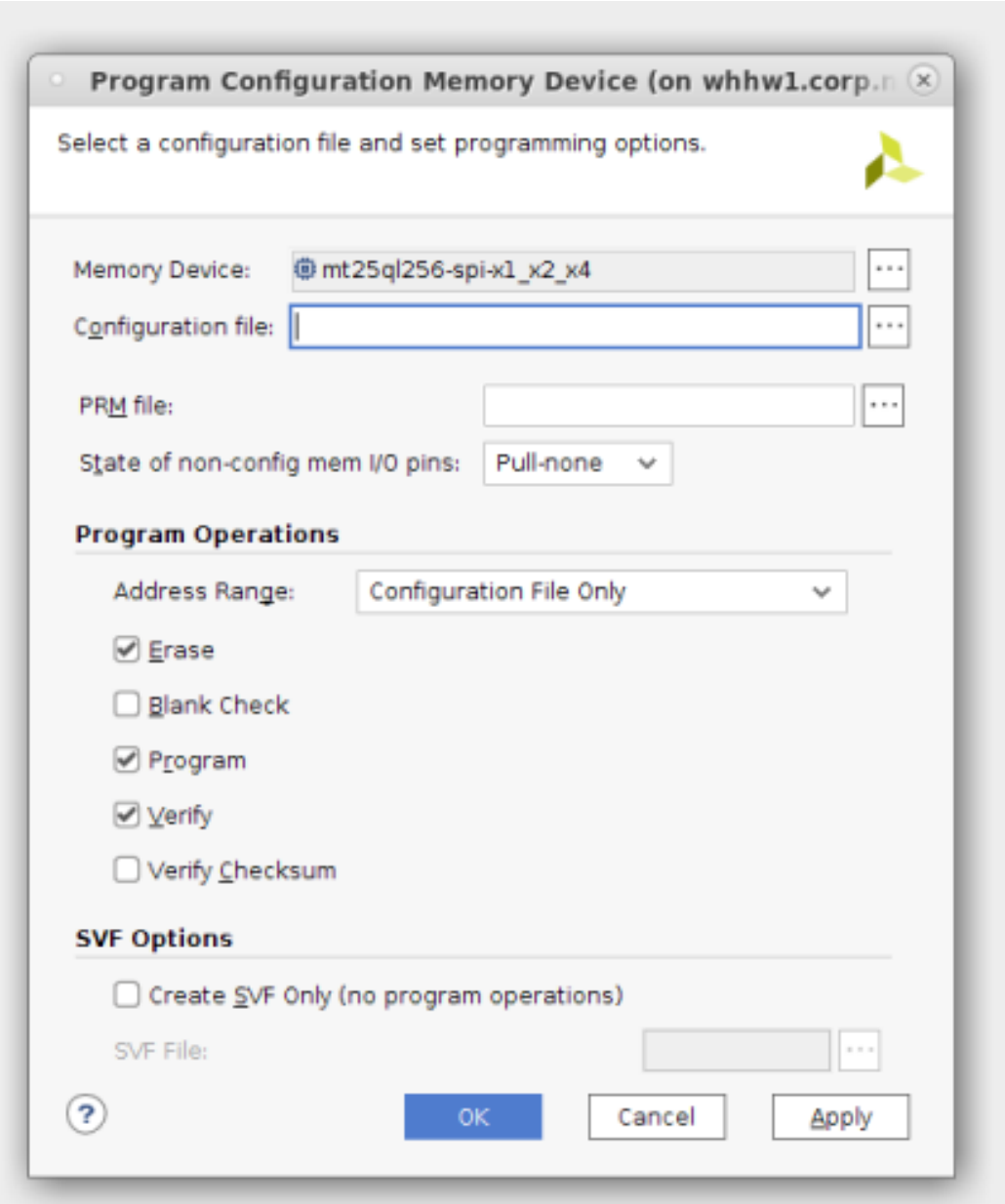

Fig. 4.30: Add the Configuration Files

5 Appendix

- **Nuclei RISC-V IP Products**: https://www.nucleisys.com/product.php
- **Nuclei Spec Documentation**: https://nucleisys.com/download.php#spec
- **Nuclei RISCV Tools and Documents**: https://nucleisys.com/download.php
- **Nuclei Prebuilt Toolchain and IDE**: https://nucleisys.com/download.php#tools
- **NMSIS**: https://github.com/Nuclei-Software/NMSIS
- **Nuclei SDK**: https://github.com/Nuclei-Software/nuclei-sdk
- **Nuclei Linux SDK**: https://github.com/Nuclei-Software/nuclei-linux-sdk
- **Nuclei Software Organization in Github**: https://github.com/Nuclei-Software/
- **RISC-V MCU Organization in Github**: https://github.com/riscv-mcu/
- **RISC-V MCU Community Website**: https://www.rvmcu.com/
- **Nuclei riscv-openocd**: https://github.com/riscv-mcu/riscv-openocd
- **Nuclei riscv-gnu-toolchain**: https://github.com/riscv-mcu/riscv-gnu-toolchain