FPGA Software Installation and Firmware Update Instructions

AMD Versal Plus Ryzen Mini-ITX Board

VPR-4616-MB

Manual Revision 1.0

July 29, 2024

SAPPHIRE - Embedded+ Initial Platform Install & Config

This page captures the steps to bring-up the Embedded+ platform configuration for the x86 Host and Versal SoC device enablement.

Hardware Setup:

Embedded+ system is comprised of a Versal SoC and a Ryzen SoC. The primary interfaces between these SoCs are PCIe and JTAG. Users interact with the Ryzen SoC through conventional means such as a keyboard and monitor, or via SSH once Linux is installed. Two UART ports are connected to the Versal SoC for debugging purposes only; they are not intended as the primary interface for Embedded+.

This documentation assumes the user possesses an Embedded+ system with OSPI pre-programmed at the factory, as this is the standard shipping configuration. The user will install Linux on the Ryzen X86 first, enabling subsequent interfacing with the Versal SoC.

x86 Host OS

This section guide user to install Linux on X86. The Embedded+ platform's X86 host XRT driver for Embedded+ has been validated with Ubuntu 22.04 OS and the XRT Ubuntu library is aligned with the GA 5.15 Linux kernel. Thus, if the default installed Ubuntu image is using a later kernel, the following steps are required to update the x86 Ubuntu OS to be aligned:

> ⓘ These steps are necessary for VPR-4616-MB only. VPR-4616-SYS has preloaded software and this section is not necessary.

1. Install the x86 host OS. Instructions and the image are available directly from Canonical:
   a Requires a USB stick, keyboard, monitor, mouse, and ethernet connection
   b Install instructions: https://ubuntu.com/tutorials/install-ubuntu-desktop#1-overview
   c OS image download: https://releases.ubuntu.com/jammy/
2. Once the x86 host OS is installed and booted from its SSD. Update the kernel to the 5.15 generic kernel with these steps
   a. Install the generic kernel

Generic kernel install
sudo apt install linux-image-generic sudo apt install -f

   b. Replace the "GRUB_DEFAULT" string in: */etc/default/grub* with *GRUB_DEFAULT="Advanced options for Ubuntu>Ubuntu, with Linux 5.15.0-###-generic".*

Note that the ### in the string above needs to be aligned with the generic kernel number used in the install of the previous step

c. Update grub config & reboot:

Kernel update
sudo update-grub sudo reboot now

3. Validate the kernel update using:

Validate kernel
uname -r

4. Install the XRT drivers on the x86 host. The Embedded+ platform support is now only be built in the 2024.1 XRT builds. This is NOT required to be aligned to the Embedded+ 2023.2 HW shell designs.

   a Get the latest XRT *xrt_202410.<date>_22.04-amd64-xrt.deb* from the automated builds at: https://www.xilinx.com/member/forms/download/xef.html?filename=xrt_202410.2.17.326_22.04-amd64-xrt.deb

   > ⓘ xrt version
   > Ensure that the XRT version is 2.17.306 or later.

   b Move xrt.deb package to the Embedded+ platform running Ubuntu 22.04

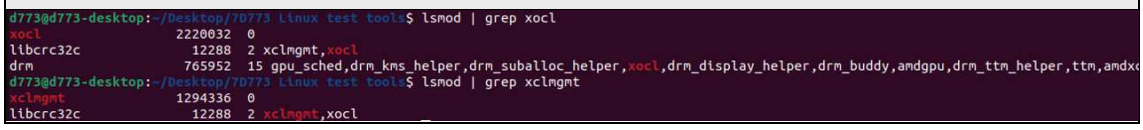
   c Install the 5.15 headers. Use the ### associated with the generic kernel installed**.**

Generic kernel headers
sudo apt install linux-headers-$(uname -r)

   d Install the xrt package with:

Install XRT driver
sudo dpkg -i xrt_202410.<date>_22.04-amd64-xrt.deb

   e The previous step may take some time as it will build the driver locally on target. After it completes verify that the drivers are installed correctly using: *lsmod*

lsmod


5. Install the Embedded+ VE2302 "base" device package

   a Get the latest base package from https://www.xilinx.com/member/forms/download/xef.html?filename=xrt-emb-plus-ve2302-base_0.5.deb

   b Move package to the Embedded+ platform.

c Install with:

Install VE2302 base design files
sudo dpkg -i xrt-emb-plus-ve2302-base_0.5.deb

6. Install the Embedded+ VE2302 XRT platform test bitstream packages
   a Get the latest test bitstream packages from:

   xrt-verify-test-ve2302_0.5.deb:
   https://www.xilinx.com/member/forms/download/xef.html?filename=xrt-verify-test-ve2302_0.5.deb

   xrt-bandwidth-dma-test-ve2302_0.5.deb
   https://www.xilinx.com/member/forms/download/xef.html?filename=xrt-bandwidth-dma-test-ve2302_0.5.deb

   xrt-aie-test-ve2302_0.5.deb
   https://www.xilinx.com/member/forms/download/xef.html?filename=xrt-aie-test-ve2302_0.5.deb

   b Move the packages to the Embedded+ platform.

   c Install with:

Install XRT test bitstreams
sudo dpkg -i xrt-verify-test-ve2302_0.5.deb
sudo dpkg -i xrt-bandwidth-dma-test-ve2302_0.5.deb
sudo dpkg -i xrt-aie-test-ve2302_0.5.deb

7. Install the Versal APU SW package
   a Get the latest APU SW package from:
   https://www.xilinx.com/member/forms/download/xef.html?filename=xrt-apu-linux-ve2302_0.5.deb

   b Move the package to the Embedded+ platform.

   c Install with:

Install Versal APU SW
sudo dpkg -i xrt-apu-linux-ve2302_0.5.deb

8. Reboot the system

On-target Self Test

The following are self-test that the user can run to test that the Versal and x86 host are set-up and configured correctly.

PCIe Connectivity

Use *lspci* to inspect if Versal device is present on PCIe bus.

lspci

```
d773@d773-desktop:~/Desktop/7D773 Linux test tools$ lspci -vd 10ee:
01:00.0 Processing accelerators: Xilinx Corporation Device 5700
        Subsystem: Xilinx Corporation Device 000e
        Flags: bus master, fast devsel, latency 0, IRQ 67, IOMMU group 9
        Memory at 1fe0000000 (64-bit, prefetchable) [size=256M]
        Memory at 1ff8040000 (64-bit, prefetchable) [size=256K]
        Capabilities: <access denied>
        Kernel driver in use: xclmgmt
        Kernel modules: xclmgmt

01:00.1 Processing accelerators: Xilinx Corporation Device 5701
        Subsystem: Xilinx Corporation Device 000e
        Flags: bus master, fast devsel, latency 0, IRQ 67, IOMMU group 10
        Memory at 1ff8000000 (64-bit, prefetchable) [size=256K]
        Memory at 1ff0000000 (64-bit, prefetchable) [size=128M]
        Capabilities: <access denied>
        Kernel driver in use: xocl
        Kernel modules: xocl
```

XRT Tests

The XRT "validate" tests are a set of PL/AIE design used to exercise basic functionality of the system. They have been installed with xrt*test*.deb packages.
Source the XRT tools:

Source XRT
source /opt/xilinx/xrt/setup.sh

Now the system is set up to run tests via xbutil command: verify, dma, mem-bw and aie. Instructions in following sections:

XRT Platform Inspection

Use XRT xbmgmt to see platform information.

XRT Platform Inspection
xbmgmt examine

xbmgmt capture

```
System Configuration
  OS Name              : Linux
  Release              : 6.5.0-41-generic
  Version              : #41~22.04.2-Ubuntu SMP PREEMPT_DYNAMIC Mon Jun  3 11:32:55 UTC 2
  Machine              : x86_64
  CPU Cores            : 4
  Memory               : 5853 MB
  Distribution         : Ubuntu 22.04.4 LTS
  GLIBC                : 2.35
  Model                :
  BIOS vendor          : American Megatrends International, LLC.
  BIOS version         : 5.24

XRT
  Version              : 2.17.326
  Branch               : 2024.1
  Hash                 : 856be14f8ad700619aa836244352b52d20f082a5
  Hash Date            : 2024-06-10 01:22:20
  XOCL                 : 2.17.326, 856be14f8ad700619aa836244352b52d20f082a5
  XCLMGMT              : 2.17.326, 856be14f8ad700619aa836244352b52d20f082a5
  Firmware Version     : N/A

Devices present
BDF             :  Shell     Logic UUID                           Device ID       Device Ready*
------------------------------------------------------------------------------------------------
[0000:01:00.0]  :  emb-plus  00000000-0000-0000-0000-000079DB078F  mgmt(inst=256)  Yes


* Devices that are not ready will have reduced functionality when using XRT tools
```

Verify Test

The "Verify" test is a simple "hello world" application for testing core ability to download a user kernel captured as an xclbin and have an expected data transfer read back from that kernel "Hello World".

Run the test:

Verify Test

```
xbutil validate -r verify -d --verbose
```

Expected output:

Verify test results

```
--------------------------------------------------------------------------------
Validation completed
Verify Test
Verbose: Enabling Verbosity
Validate Device           : [0000:01:00.1]
    Platform              : emb-plus
    SC Version            : 0.0.0
    Platform ID           : 00000000-0000-0000-0000-000079DB078F
--------------------------------------------------------------------------------
Test 1 [0000:01:00.1]     : verify
    Description           : Run 'Hello World' kernel test
    Test Status           : [PASSED]
--------------------------------------------------------------------------------
```

DMA Test

The "DMA" test is a simple DMA test that transfer data between Versal and Ryzen using DMA on Versal.

Run the test:

DMA test
xbutil validate -r dma -d --verbose

Expected output:

DMA test results

```
Verbose: Enabling Verbosity
Validate Device           : [0000:01:00.1]
    Platform              : emb-plus
    SC Version            : 0.0.0
    Platform ID           : 00000000-0000-0000-0000-000079DB078F
-------------------------------------------------------------------------------
st 1 [0000:01:00.1]       : dma
    Description           : Run dma test
    Details               : Buffer size - '16 MB' Memory Tag - 'MC_NOC'
                            Host -> PCIe -> FPGA write bandwidth = 2704.1 MB/s
                            Host <- PCIe <- FPGA read bandwidth = 3504.2 MB/s
    Test Status           : [PASSED]
-------------------------------------------------------------------------------
```

Bandwidth Test

The "bandwidth" test runs a limited bandwidth test on DDR memory and PCIe data transfers.

Run the test:

Bandwidth test
xbutil validate -r mem-bw -d --verbose

Expected output:

Bandwidth test results

```
-------------------------------------------------------------------------------
Validation completed
Bandwith Test
Verbose: Enabling Verbosity
Validate Device           : [0000:01:00.1]
    Platform              : emb-plus
    SC Version            : 0.0.0
    Platform ID           : 00000000-0000-0000-0000-000079DB078F
-------------------------------------------------------------------------------
Test 1 [0000:01:00.1]     : mem-bw
    Description           : Run 'bandwidth kernel' and check the throughput
    Details               : Throughput (Type: DDR) (Bank count: 1) : 19002.7 MB/s
                            Throughput of Memory Tag: MC_NOC : 19002.7 MB/s
    Test Status           : [PASSED]
-------------------------------------------------------------------------------
```

AIE Test

The "aie" test runs a AIE tile functionality test.

Run the test:

AIE test
xbutil validate -r aie -d --verbose

Expected output:

AIE test results

```
----------------------------------------------------------------------------
Validation completed
AIE Test
Verbose: Enabling Verbosity
Validate Device           : [0000:01:00.1]
    Platform              : emb-plus
    SC Version            : 0.0.0
    Platform ID           : 00000000-0000-0000-0000-000079DB078F
----------------------------------------------------------------------------
Test 1 [0000:01:00.1]     : aie
    Description           : Run AIE PL test
    Test Status           : [PASSED]
----------------------------------------------------------------------------
```

Debug Tools – Versal Serial Console

The Versal serial console is connected to the Ryzen device on the motherboard. Therefore user can access the uart outputs from Ryzen. In Ubuntu, first download picocom:

Install picocom
sudo apt-get install picocom

Then user can access the com ports on commandline from Ubuntu:

Connect to APU serial output:
sudo picocom -b 115200 /dev/ttyUSB1

Connect to PLM/RPU serial output:
sudo picocom -b 115200 /dev/ttyUSB2

FPGA(Xilinx) F/W Update

The VPR-4616-MB and VPR-4616-SYS are both shipped with OSPI image. You can check their version with "xbmgmt examine" command. A UUID of 00000000-0000-0000-0000-000079DB078F corresponds to emb-plus-ospi-emb-plus-ve2302-20240620051522.bin (0.5 release). Upgrading OSPI is necessary if the UUID does not match the release.

Software requirements

1. Vivado Installed
2. FPGA(Xilinx) F/W: Versal OSPI image file (*.bin) and PDI file (*.pdi)

PDI = Programmable Device Image

The boot image for Versal devices is called a PDI, which is an AMD file format processed by the PLM as part of the boot process or partial configuration process. The PDI data is specific to the design requirements. The boot image for Versal devices typically involves binaries used to boot and configure the platform.
It is stored in OSPI flash.

Update Instruction:

Install Vivadolab

1. Install Vivado_lab 2023.2

   a Download Vivado_lab for Linux from https://www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/vivado-design-tools/2023-2.html, Xilinx instructions can be found in UG973 (https://docs.amd.com/r/en-US/ug973-vivado-release-notes-install-license)

   b Extract the downloaded file. In the folder, give executable permission to installLibs.sh and xsetup and then execute:

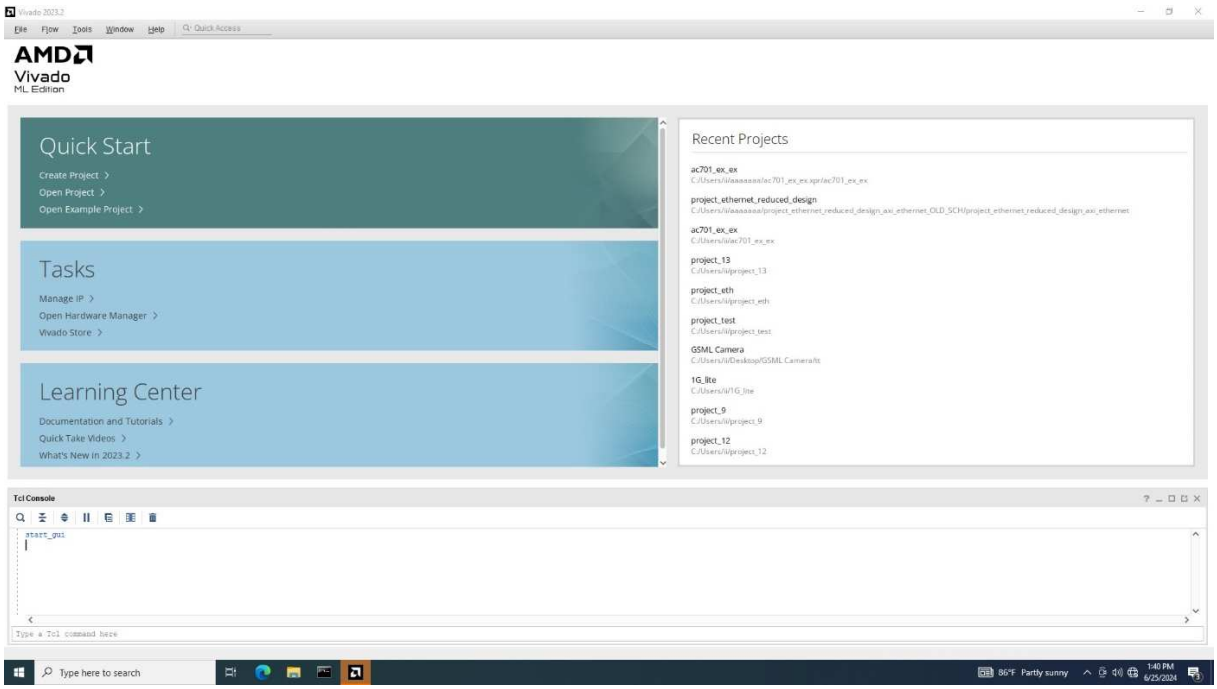
```
chmod +x installLibs.sh
sudo ./installLibs.sh
chmod +x xsetup
sudo ./xsetup
(if any libraries are missing – install them manually, follow GUI to finish installing Vivado_lab )
cd ${vivado_install_dir}/data/xicom/cable_drivers/lin64/install_script/install_drivers/
sudo ./install_drivers
sudo reboot now
```

c To launch Vivado_lab:

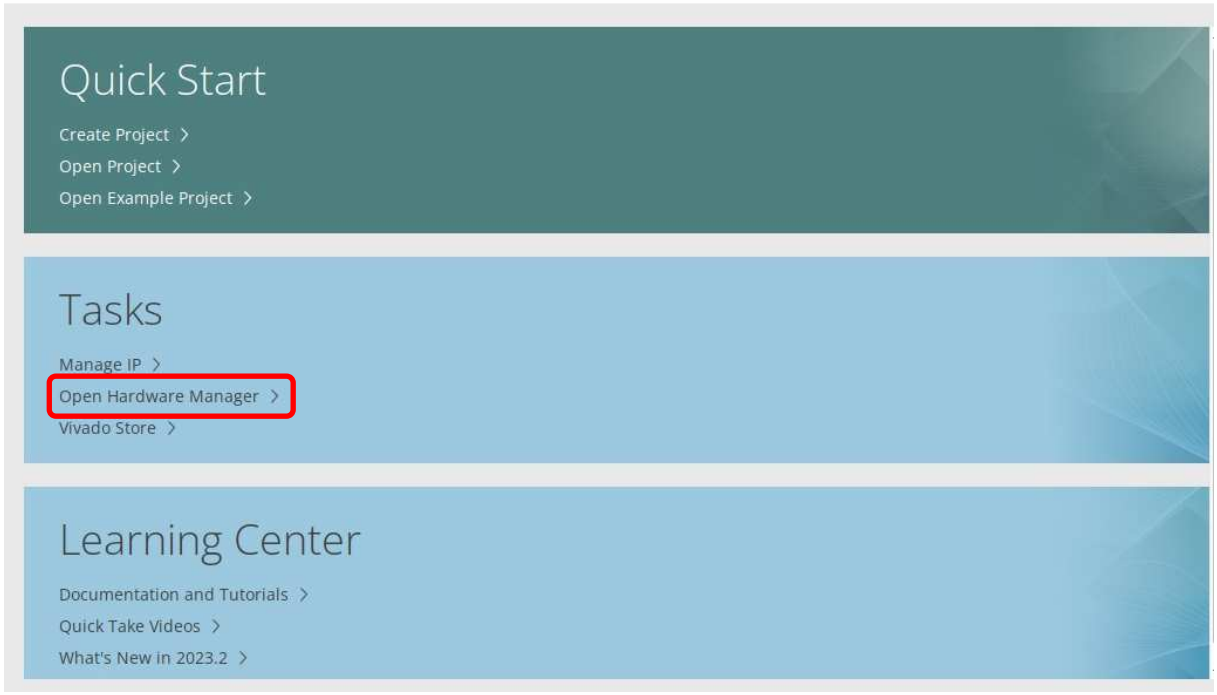
```
source $(vivado_install_dir)/settings64.sh
vivado_lab
```

Note: $(vivado_install_dir) is /tools/Xilinx/Vivado_Lab/2023.2 by default

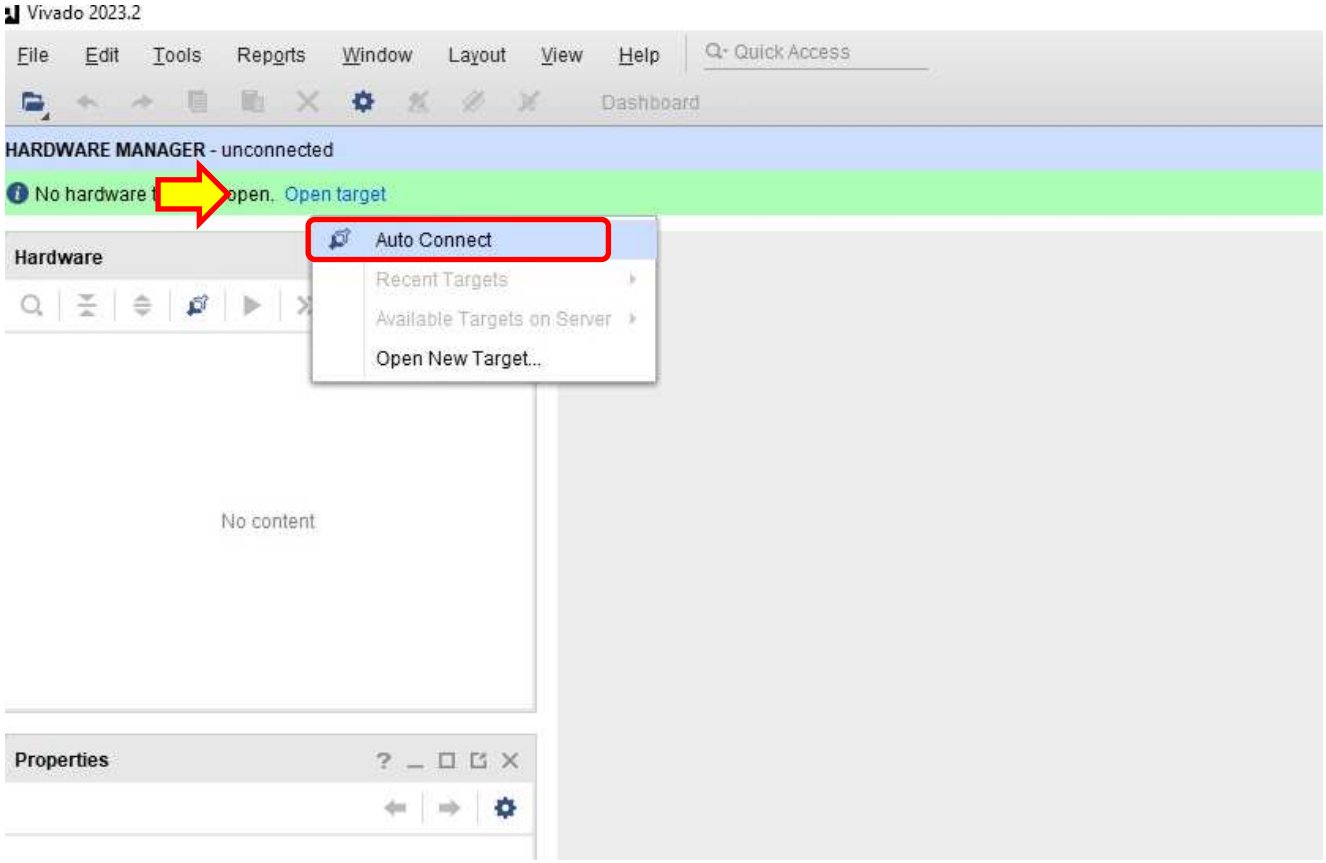
Open Vivado



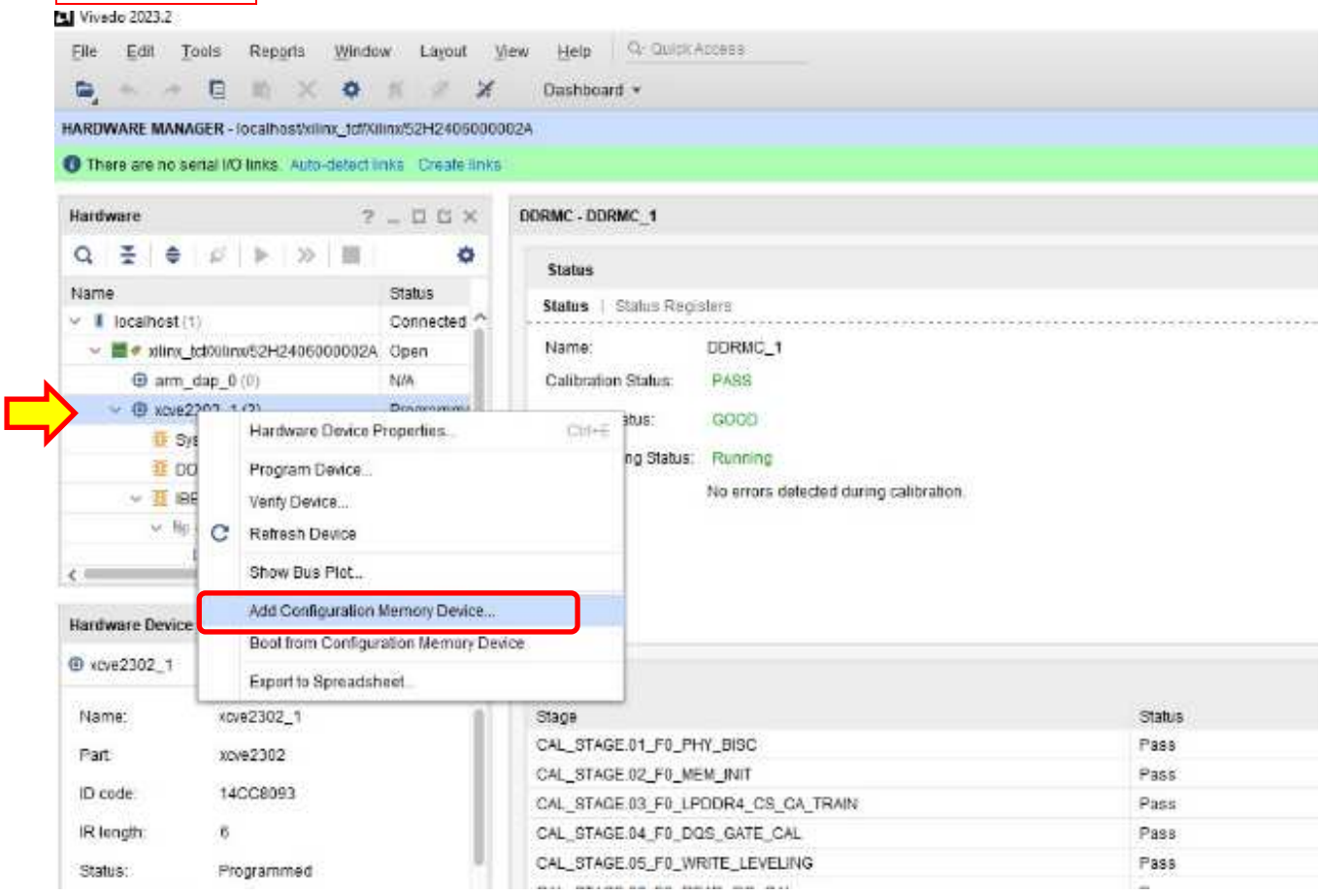
1. Click Open Hardware Manager



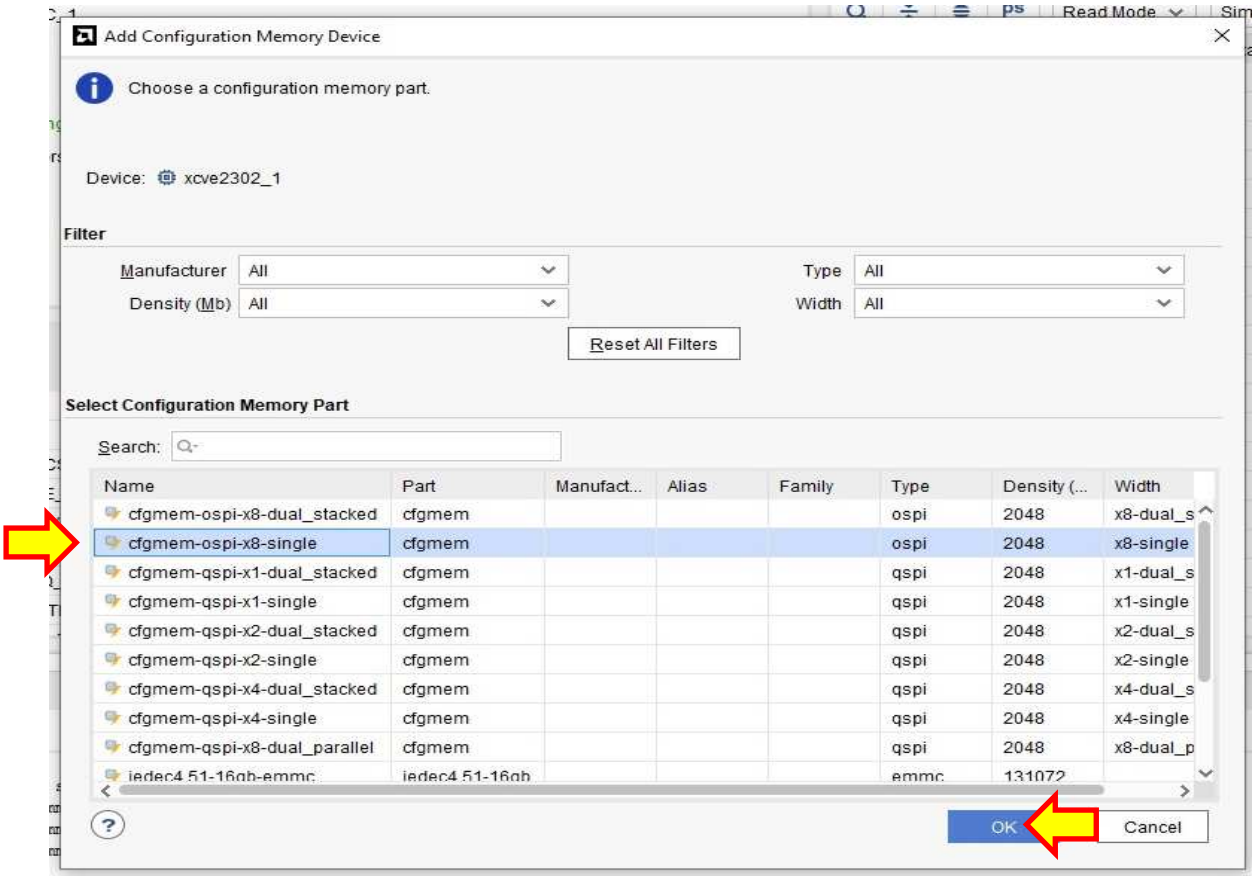
2. Click Open target and select Auto Connect



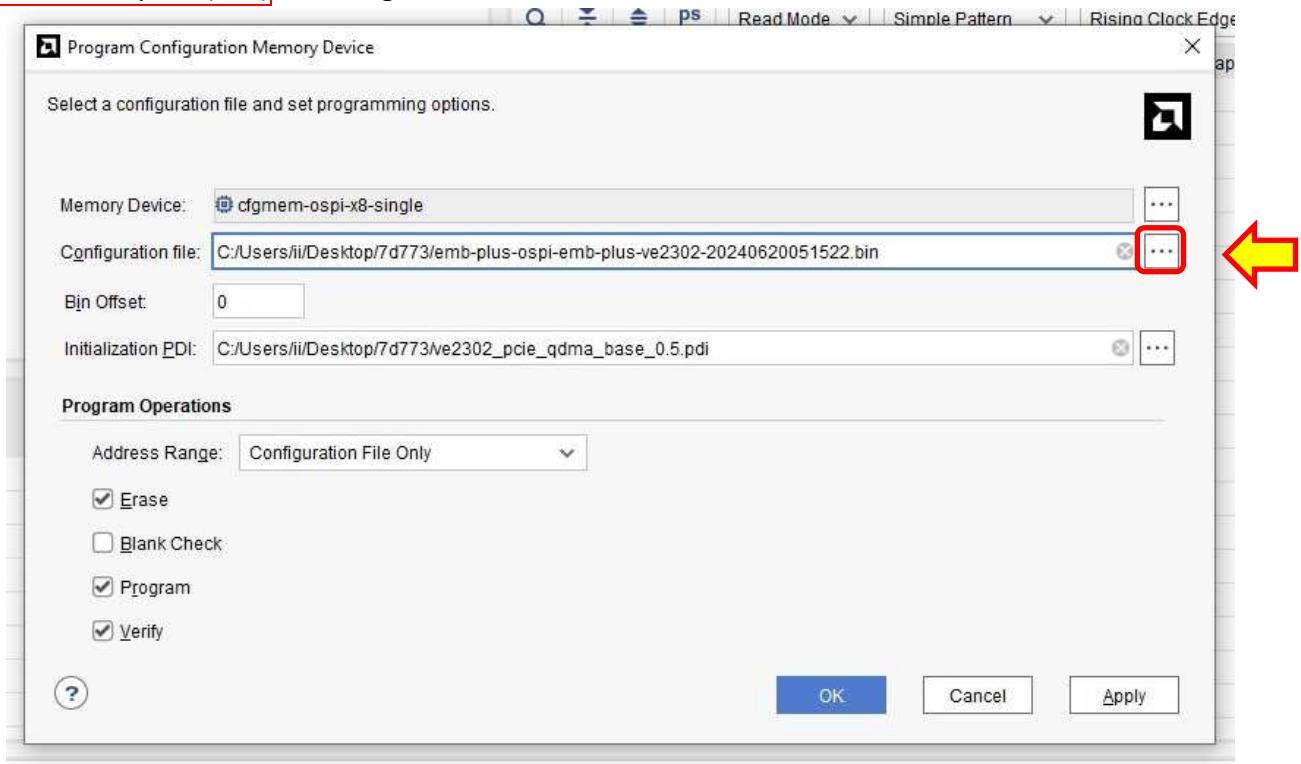
3. Move the cursor to xcve2302_1, press the right button of the mouse, and select Add Configuration Memory Device



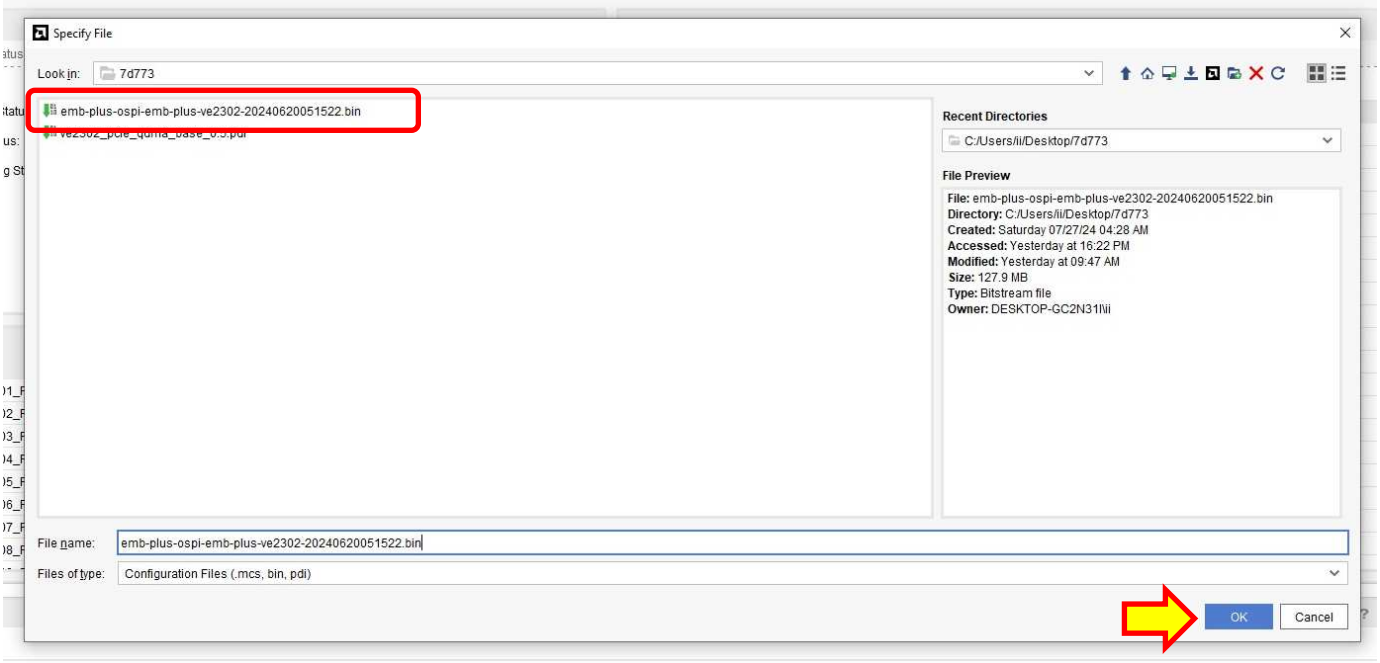
4. Select Cfgmem-ospi-x8-single then click OK



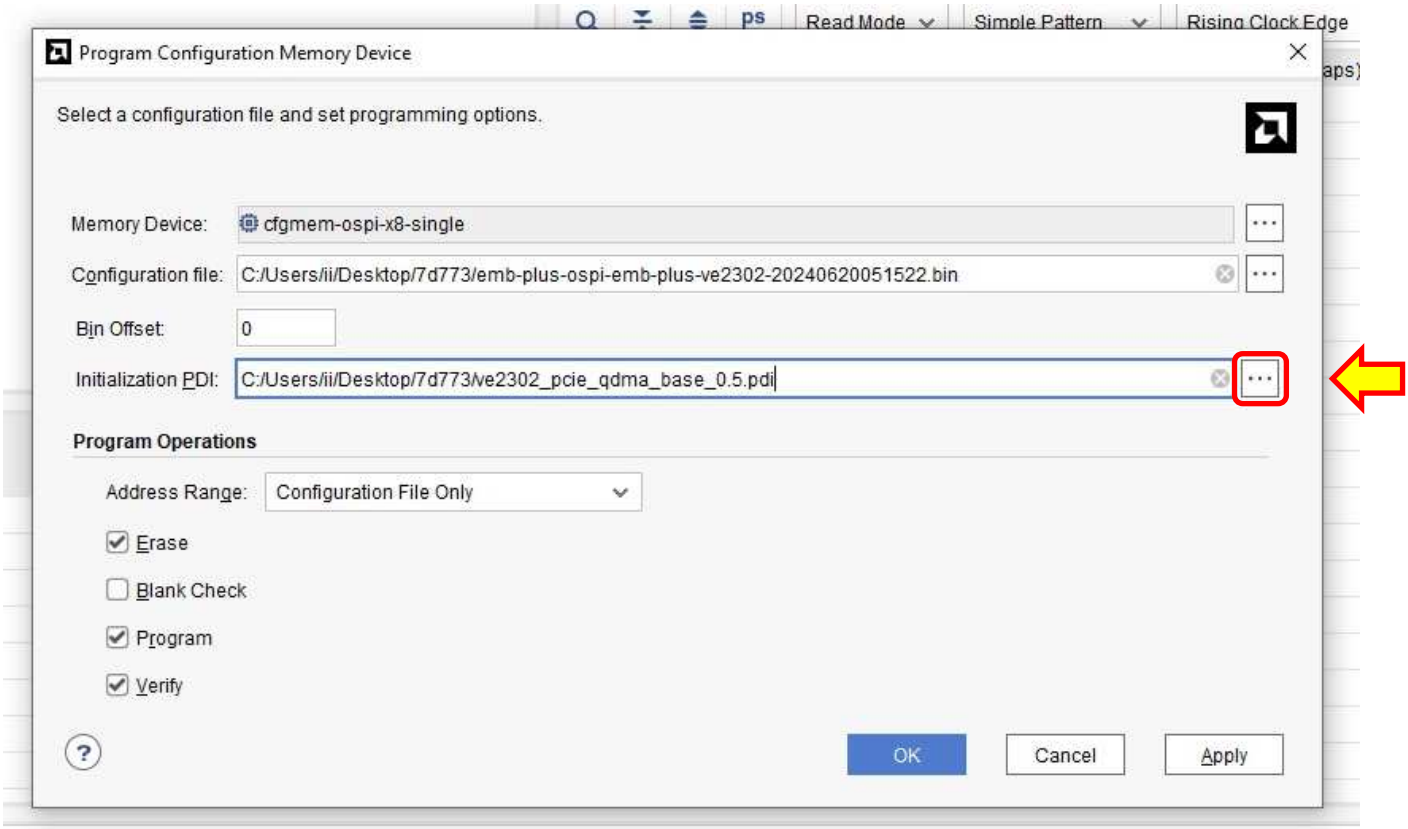
5. Selected file path (BIN) of Configuration file



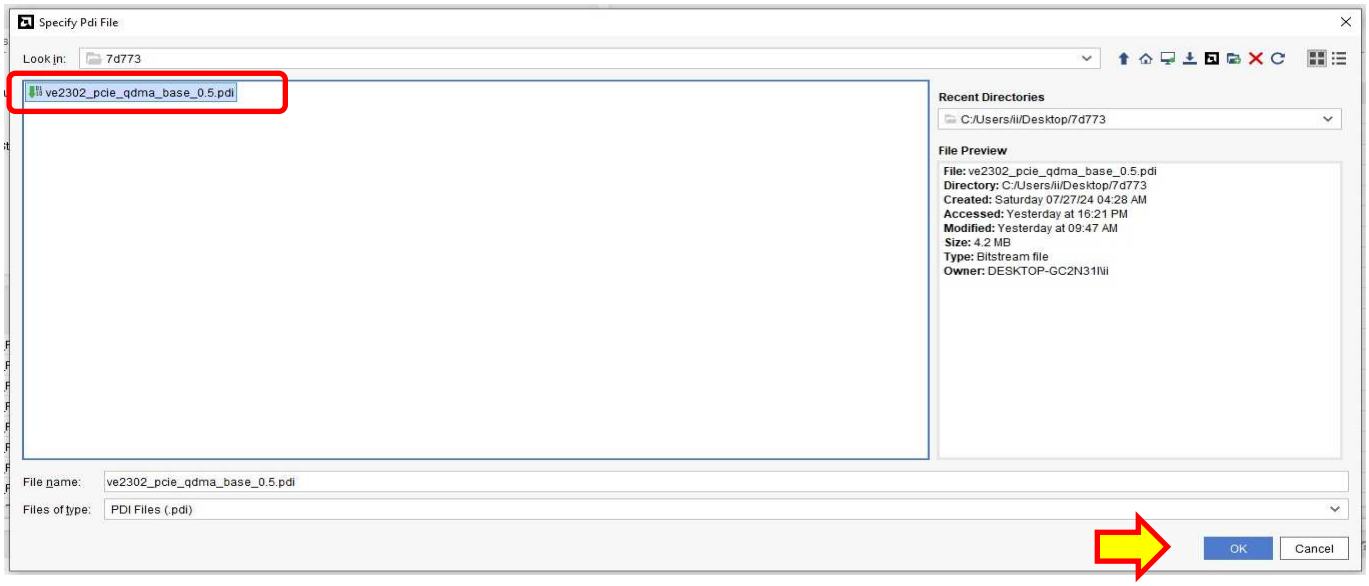
6. Select emb-plus-ospi-emb-plus-ve2302-20240620051522.bin and click OK



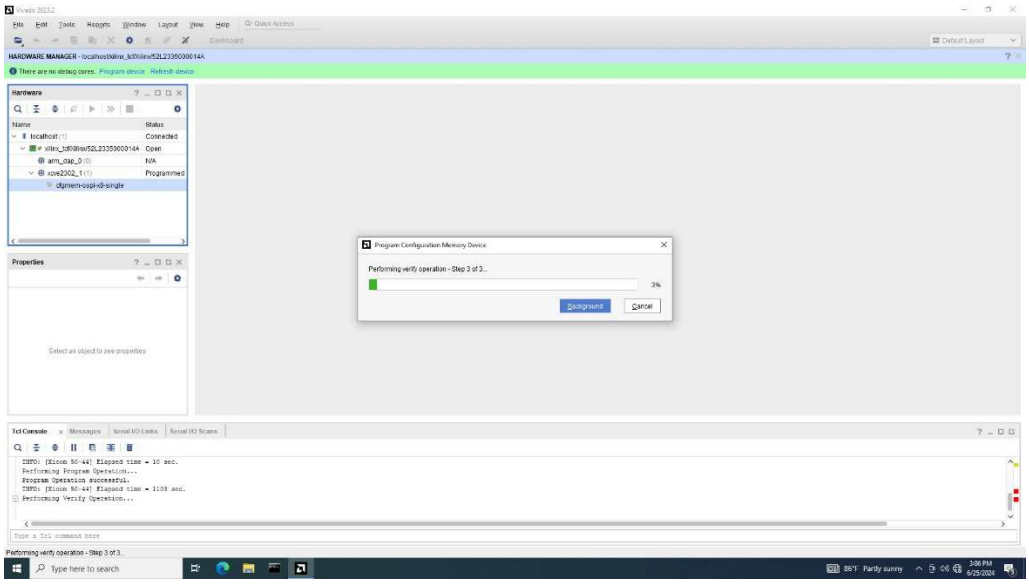
7. Selected file path (PDI) of Initialization PDI



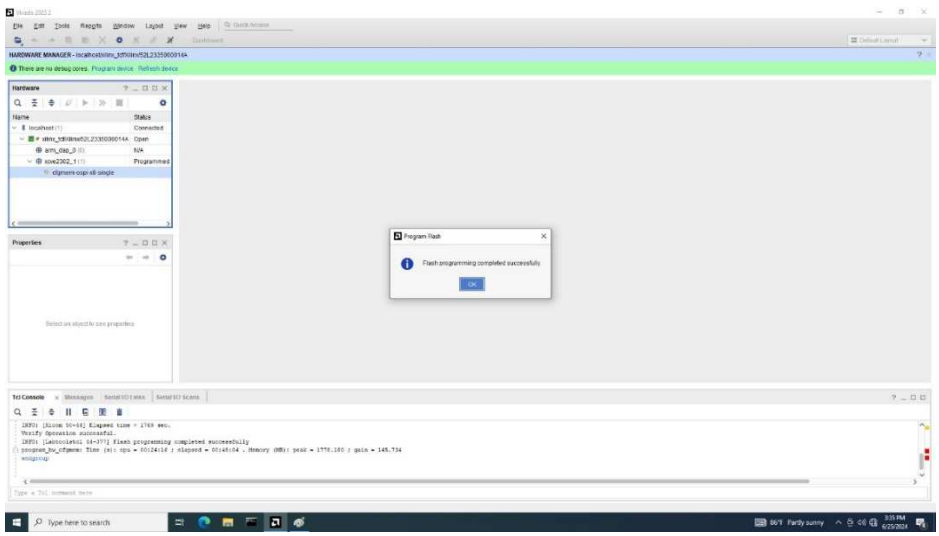
8. Select ve2302_pcie_qdma_base_0.5.pdi and click OK, Programming starts after OK is clicked.



9. Programming…(Approximate programming time is 50 mins)



10. Programmed OK



11. Reboot the system