# Use ROCm™ on Radeon™ GPUs Documentation

Advanced Micro Devices, Inc.

May 06, 2024

# CONTENTS

**Turn your desktop into a Machine Learning platform with the latest high-end AMD Radeon™ 7000 series GPUs**

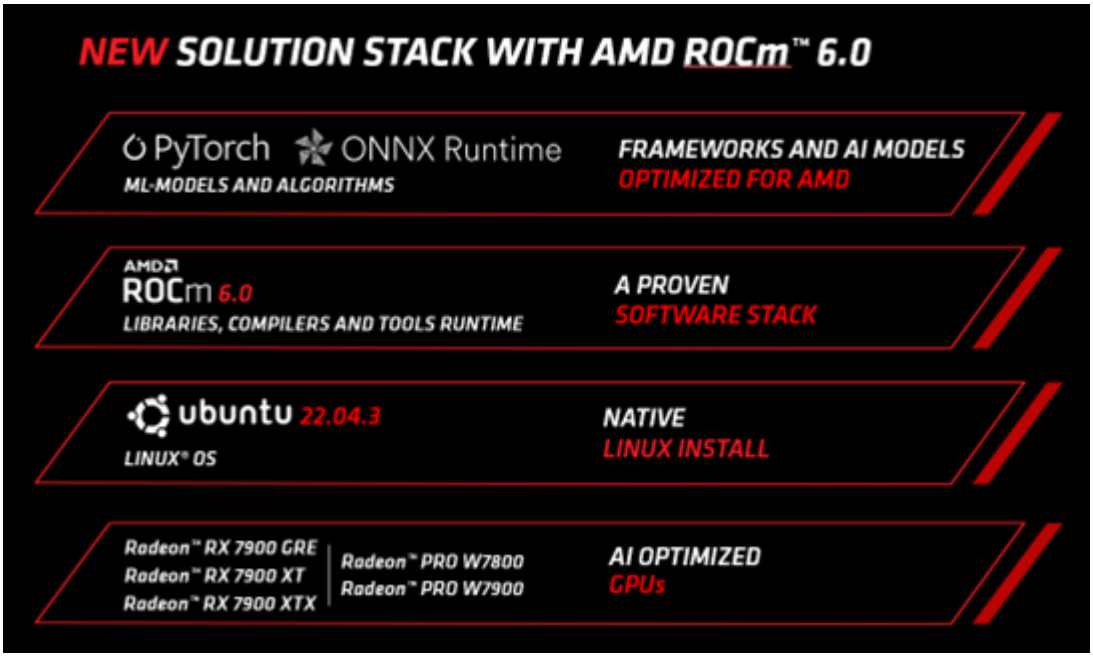
AMD has expanded support for Machine Learning Development on RDNA™ 3 GPUs with Radeon™ Software for Linux 23.40.2 with ROCm™ 6.0!

Researchers and developers working with Machine Learning (ML) models and algorithms using PyTorch and ONNX Runtime can now also use ROCm 6.0 on Linux® to tap into the parallel computing power of the latest high-end AMD Radeon 7000 series desktop GPUs, and based on AMD RDNA 3 GPU architecture.

A client solution built on powerful high-end AMD GPUs enables a local, private, and cost-effective workflow to develop ROCm and train Machine Learning for users who were solely reliant on cloud-based solutions.

**More ML performance for your desktop**

- With today's models easily exceeding the capabilities of standard hardware and software not designed for AI, ML engineers are looking for cost-effective solutions to develop and train their ML-powered applications. Due to the availability of significantly large GPU memory sizes of 24GB or 48GB, utilization of a local PC or workstation equipped with the latest high-end AMD Radeon 7000 series GPU offers a robust/potent yet economical option to meet these expanding ML workflow challenges.
- Latest high-end AMD Radeon 7000 series GPUs are built on the RDNA 3 GPU architecture,
  - featuring more than 2x higher AI performance per Compute Unit (CU) compared to the previous generation
  - now comes with up to 192 AI accelerators
  - offers up to 24GB or 48GB of GPU memory to handle large ML models




**Note:** Based on AMD internal measurements, November 2022, comparing the Radeon RX 7900 XTX at 2.505 GHz boost clock with 96 CUs issuing 2X the Bfloat16 math operations per clock vs. the Radeon RX 6900 XT GPU at 2.25 GHz boost clock and 80 CUs issue 1X the Bfloat16 math operations per clock. Results may vary. RX-821.

**Migrate your application from the desktop to the datacenter**

- ROCm is the open-source software stack for Graphics Processing Unit (GPU) programming. ROCm spans several domains: General-Purpose computing on GPUs (GPGPU), High Performance Computing (HPC) and heterogeneous computing.

- The latest AMD ROCm 6.0 software stack for GPU programming unlocks the massively parallel compute power of these RDNA 3 GPUs for use with various ML frameworks. The same software stack also supports AMD CDNA™ GPU architecture, so developers can migrate applications from their preferred framework into the datacenter.

**Freedom to customize**

ROCm is primarily Open-Source Software (OSS) that allows developers the freedom to customize and tailor their GPU software for their own needs while collaborating with a community of other developers, and helping each other find solutions in an agile, flexible, rapid and secure manner. AMD ROCm allows users to maximize their GPU hardware investment. ROCm is designed to help develop, test and deploy GPU accelerated HPC, AI, scientific computing, CAD, and other applications in a free, open-source, integrated and secure software ecosystem.

**Improved interoperability**

- Support for PyTorch, one of the leading ML frameworks.
- Support for ONNX Runtime to perform inference on a wider range of source data, including INT8 with MIGraphX.

---

**Note:** Visit AMD ROCm Documentation for the latest on ROCm.

For the latest driver installation packages, visit Linux Drivers for Radeon Software.

---

CHAPTER

ONE

# PREREQUISITES TO USE ROCM ON RADEON DESKTOP GPUS FOR MACHINE LEARNING DEVELOPMENT

Before starting with the installation, ensure that your system meets the necessary requirements such as supported hardware, a compatible operating system, and the recommended system configuration to ensure optimal performance and functionality.

See *Compatibility matrices* for more information.

## 1.1 Supported hardware

### 1.1.1 Supported graphics processing units

To successfully install ROCm™ for machine learning development, ensure that your system is operating on a Radeon™ Desktop GPU listed in the *Compatibility matrices* section.

### 1.1.2 Recommended memory

The recommended memory to use ROCm on Radeon. These specifications are required for complex AI/ML workloads:

- 64GB Main Memory
- 24GB GPU Video Memory

#### Minimum recommendations

Minimum memory requirements to use ROCm on Radeon. Note that low system memory may cause issues running inference models on CPU.

- 16GB Main Memory
- 8GB GPU Video Memory

## 1.2 Supported operating systems

Ensure that your operating system is up-to-date to successfully install ROCm for machine learning development.

Refer to *Compatibility matrices* for up-to-date operating system compatibility.

### 1.2.1 Update Ubuntu® operating system

Use the following commands to bring your OS up-to-date:

```
sudo apt-get update
sudo apt-get dist-upgrade
```

## 1.3 Recommended system configuration

This section guides users on how to optimize system configuration for ROCm™ usage, ensuring smooth and performant ROCm operation.

### 1.3.1 Disable iGPU

The iGPU is non-essential for AI and ML workloads and not officially supported. Disable iGPU in SBIOS before proceeding to avoid unknown issues.

Alternatively, use environment variables to select the target GPU.

Here are examples to disable iGPU on some AMD motherboards:

#### Gigabyte™ X670 AORUS ELITE AX

1. Enter BIOS

   **Path**: *Advanced → AMD CBS → NBIO Common Options → GFX Configuration → iGPU Configuration*

2. Set iGPU to `Disabled`

#### ASUS Prime X670-P WIFI

1. Enter BIOS

   **Path**: *Advanced → NB Configuration → Integrated Graphics*

2. Set to `Disabled`

   **NOTE:** This step only applies to AMD motherboards, no action is required for non-AMD motherboards.

   There are no minimum motherboard hardware requirements.

### Alternative option: Use environment variables to select target GPU

An alternative option to disabling the iGPU is to use environment variable to select the GPU.

See GPU Isolation Techniques to specify the device indices you would like to expose to your application.

CHAPTER

TWO

# HOW TO GUIDE - USE ROCM ON RADEON GPUS

This guide walks you through the various installation processes required to pair ROCm™ with the latest high-end AMD Radeon™ 7000 series desktop GPUs, and get started on a fully-functional environment for AI and ML development.

## 2.1 Install Radeon software for Linux with ROCm

The ROCm™ Software Stack and other Radeon™ software for Linux components are installed using the amdgpu-install script to assist you in the installation of a coherent set of stack components.

- Simplifies the installation of the AMDGPU stack by encapsulating the distribution specific package installation logic and by using command line options that allows you to specify the:
  - Usecase of the AMDGPU stack to be installed (Graphics or Workstation)
  - Combination of components (Pro stack, or user selection)
- Performs post-install checks to verify whether the installation was performed successfully.
- Installs the uninstallation script to allow you to remove the whole AMDGPU stack from the system by using a single command.

The script is provided by the installer package. See *Compatibility matrices* for support information.

### 2.1.1 Install AMD unified driver package repositories and installer script

Download and install the amdgpu-install script on the system.

Enter the following commands to install the installer script for the latest compatible Ubuntu® version:

```
sudo apt update
wget http://repo.radeon.com/amdgpu-install/23.40.2/ubuntu/jammy/amdgpu-install_6.0.
↪60002-1_all.deb
sudo apt install ./amdgpu-install_6.0.60002-1_all.deb
```

### 2.1.2 Install AMD unified kernel-mode GPU driver, ROCm, and graphics

After the Unified Driver Deb Package repositories are installed, run the installer script with appropriate `--usecase` parameters to install the driver components.

AMD recommends installing the Graphics usecase by default. Only consider the alternative install option if you have an applicable Workstation usecase scenario.

Enter the following command to display a list of available usecases:

```
sudo amdgpu-install --list-usecase
```

#### Option A: Graphics usecase

AMD recommends installing the Graphics usecase by default.

1. Run the following command to install open source graphics and ROCm.

   ```
   amdgpu-install -y --usecase=graphics,rocm
   ```

   Watch for output warning or errors indicating an unsuccessful driver installation.

   **NOTE:** The `-y` option installs non-interactively. This step may take several minutes, depending on internet connection and system speed.

2. Reboot the system.

   ```
   sudo reboot
   ```

   See Using the amdgpu-install script for more information.

#### Option B: Workstation usecase

Only consider the alternative install option if you have an applicable Workstation usecase scenario.

1. Run the following command to install workstation graphics and ROCm.

   ```
   amdgpu-install -y --usecase=workstation,rocm
   ```

   **NOTE:** The -y option installs non-interactively. This step may take several minutes, depending on internet connection and system speed.

2. Reboot the system.

   ```
   sudo reboot
   ```

   See Using the amdgpu-install script for more information.

### Set permissions for Groups to allow access to GPU hardware resources

Once the driver is installed, add any current user to the render and video groups to access GPU resources.

Reboot in order for group changes to take effect.

### Add user to render and video groups

1. Enter the following command to check groups in the system:

```
groups
```

2. Add user to the render and video group using the command:

```
sudo usermod -a -G render,video $LOGNAME
```

3. Reboot the system.

```
sudo reboot
```

See Setting Permissions for Groups for more information.

### Post-install verification checks

Run these post-installation checks to verify that the installation is complete:

1. Verify that the current user is added to the render and video groups.

```
groups
```

**Expected result**:

```
<username> adm cdrom sudo dip video plugdev render lpadmin lxd sambashare
```

`<username>` indicates the current user, and this result will vary in your environment.

2. Check if *amdgpu* kernel driver is installed.

```
dkms status
```

**Expected result**:

```
amdgpu/x.x.x-xxxxxxx.xx.xx, x.x.x-xx-generic, x86_64: installed
```

3. Check if the GPU is listed as an agent.

```
rocminfo
```

**Expected result**:

```
[...]
*******
Agent 2
*******
  Name:                    gfx1100
  Uuid:                    GPU-5ecee39292e80c37
```

(continues on next page)

(continued from previous page)

```
  Marketing Name:          Radeon RX 7900 XTX
  Vendor Name:             AMD
  [...]
[...]
```

4. Check if the GPU is listed.

```
clinfo
```

**Expected result**:

```
[...]
  Platform Name:                 AMD Accelerated Parallel Processing
Number of devices:               1
  Device Type:                   CL_DEVICE_TYPE_GPU
  Vendor ID:                     1002h
  Board name:                    Radeon RX 7900 XTX
[...]
```

See Installing the all open usecase for additional troubleshooting tips.

### 2.1.3 Advanced install methods

For advanced install methods, such as Multi-Version and Package Manager, refer to AMD GPU Install Script.

### 2.1.4 Uninstall ROCm

Run the following command to uninstall the ROCm software stack and other Radeon software for Linux components:

```
sudo amdgpu-uninstall
```

### 2.1.5 Upgrade to newer versions of Radeon software for Linux

The recommended method to upgrade is to uninstall, followed by an install.

Radeon Software for Linux does not support in-place upgrades.

## 2.2 Install PyTorch for ROCm

Refer to this section for the recommended PyTorch via PIP installation method, as well as Docker-based installation.

### Option A: PyTorch via PIP installation

AMD recommends the PIP install method to create a PyTorch environment when working with ROCm™ for machine learning development.

> Check Pytorch.org for latest PIP install instructions and availability. See *Compatibility matrices* for support information.

### Install PyTorch via PIP

1. Enter the following command to unpack and begin set up.

```
sudo apt install python3-pip -y
```

2. Enter this command to update the pip wheel.

```
pip3 install --upgrade pip wheel
```

3. Enter this command to install Torch and Torchvision for ROCm AMD GPU support.

```
wget https://repo.radeon.com/rocm/manylinux/rocm-rel-6.0.2/torch-2.1.2+rocm6.0-
↪cp310-cp310-linux_x86_64.whl
wget https://repo.radeon.com/rocm/manylinux/rocm-rel-6.0.2/torchvision-0.16.
↪1+rocm6.0-cp310-cp310-linux_x86_64.whl
pip3 install --force-reinstall torch-2.1.2+rocm6.0-cp310-cp310-linux_x86_64.whl␣
↪torchvision-0.16.1+rocm6.0-cp310-cp310-linux_x86_64.whl
```

> This may take several minutes.
>
> **Important!** AMD recommends proceeding with ROCm WHLs available at repo.radeon.com. The ROCm WHLs available at PyTorch.org are not tested extensively by AMD as the WHLs change regularly when the nightly builds are updated.

Next, *verify your PyTorch installation*.

### Option B: Docker installation

Using Docker provides portability, and access to a prebuilt Docker container that has been rigorously tested within AMD. Docker also cuts down compilation time, and should perform as expected without installation issues.

### Prerequisites to install PyTorch using Docker

Docker for Ubuntu® must be installed.

To install Docker for Ubuntu, enter the following command:

```
sudo apt install docker.io
```

#### Use Docker image with pre-installed PyTorch

Follow these steps for installing using a Docker image.

1. Enter the following command to pull the public PyTorch Docker image.

```
sudo docker pull rocm/pytorch:rocm6.0.2_ubuntu22.04_py3.10_pytorch_2.1.2
```

   **Optional:** You can also download a specific and supported configuration with different user-space ROCm versions, PyTorch versions, and supported operating systems.

   Refer to hub.docker.com/r/rocm/pytorch to download the PyTorch Docker image.

2. Start a Docker container using the downloaded image.

```
sudo docker run -it \
  --cap-add=SYS_PTRACE \
  --security-opt seccomp=unconfined \
  --device=/dev/kfd \
  --device=/dev/dri \
  --group-add video \
  --ipc=host \
  --shm-size 8G \
  rocm/pytorch:rocm6.0.2_ubuntu22.04_py3.10_pytorch_2.1.2
```

   This will automatically download the image if it does not exist on the host. You can also pass the `-v` argument to mount any data directories from the host onto the container.

Next, *verify your PyTorch installation*.

See PyTorch Installation for ROCm for more information.

### 2.2.1 Verify PyTorch installation

Confirm if PyTorch is correctly installed.

1. Verify if Pytorch is installed and detecting the GPU compute device.

```
python3 -c 'import torch' 2> /dev/null && echo 'Success' || echo 'Failure'
```

   **Expected result**:

```
Success
```

2. Enter command to test if the GPU is available.

```
python3 -c 'import torch; print(torch.cuda.is_available())'
```

   **Expected result**:

```
True
```

3. Enter command to display installed GPU device name.

```
python3 -c "import torch; print(f'device name [0]:', torch.cuda.get_device_
↪name(0))"
```

   **Expected result**: Example: *device name [0]: Radeon RX 7900 XTX*

```
device name [0]: <Supported AMD GPU>
```

4. Enter command to display component information within the current PyTorch environment.

```
python3 -m torch.utils.collect_env
```

**Expected result**:

```
PyTorch version

ROCM used to build PyTorch

OS

Is CUDA available

GPU model and configuration

HIP runtime version

MIOpen runtime version
```

Environment set-up is complete, and the system is ready for use with PyTorch to work with machine learning models, and algorithms.

## 2.3 Install ONNX Runtime for Radeon GPUs

Refer to this section to install ONNX via the PIP installation method.

### 2.3.1 Overview

Ensure that the following prerequisite installations are successful before proceeding to install ONNX Runtime for use with ROCm™ on Radeon™ GPUs.

#### Prerequisites

- Radeon Software for Linux (with ROCm) is installed.
- *MIGraphX* is installed. This enables ONNX Runtime to build the correct MIGraphX EP.
- The half library is installed.

  **NOTE** The half library is installed with MIGraphX. If the half library install is not verified, use the following install command:

  ```
  $ sudo apt install half
  ```

- If the prerequisite installations are successful, proceed to *install ONNX Runtime*.

  **NOTE** Unless adding custom features, use the pre-built Python wheel files provided in the PIP installation method.

### 2.3.2 Install ONNX Runtime

**Important!**

- Use the provided pre-built Python wheel files from the PIP installation method, unless adding custom features.
- The wheel file contains the MIGraphX and ROCm Execution Providers (EP). Refer to *Install MIGraphX for ONNX RT* for more information.
- Refer to ONNX Runtime Documentation for additional information on ONNX Runtime topics.
- See ONNX Runtime Tutorials to try out real applications and tutorials on how to get started.

#### ONNX Runtime install via PIP installation method

Use the PIP install method to create an ONNX Runtime environment.

**To install via PIP**,

Enter this command to download and install the ONNX Runtime wheel.

```
pip3 install https://repo.radeon.com/rocm/manylinux/rocm-rel-6.0.2/onnxruntime_rocm-
↪inference-1.17.0-cp310-cp310-linux_x86_64.whl
```

Next, *verify your ONNX Runtime installation*.

#### Verify ONNX Runtime installation

Confirm if ONNX Runtime is correctly installed.

```
$ python3
$ import onnxruntime as ort
  ort.get_available_providers()
```

**Expected result**: The following EPs are displayed.

```
>>> ort.get_available_providers()
['MIGraphXExecutionProvider', 'ROCMExecutionProvider', 'CPUExecutionProvider']
```

Environment set-up is complete, and the system is ready for use with ONNX Runtime to work with machine learning models, and algorithms.

## 2.4 Install MIGraphX for Radeon GPUs

MIGraphX is AMD's graph inference engine that accelerates machine learning model inference, and can be used to accelerate workloads within the Torch MIGraphX and ONNX Runtime backend frameworks.

- Torch-MIGraphX, which integrates MIGraphX with PyTorch
- MIGraphX for ONNX Runtime backend, which integrates MIGraphX with ONNX

  ONNX Runtime can be driven by either the ROCm™ Execution Provider (EP) or MIGraphX EP

### 2.4.1 Introduction to MIGraphX

MIGraphX is a graph optimizer that accelerates the inference for deep learning models. It provides C++ and Python APIs that are integrated within frameworks like Torch MIGraphX, ONNX Runtime, and other user solutions. The following process summarizes the procedures that occur under-the-hood during the optimization and real-time compilation process.

MIGraphX accelerates the Machine Learning models by leveraging several graph-level transformations and optimizations. These optimizations include:

- Operator fusion
- Arithmetic simplifications
- Dead-code elimination
- Common subexpression elimination (CSE)
- Constant propagation

When the aforementioned optimizations are applied, MIGraphX emits code for the AMD GPU by calling to MIOpen, rocBLAS, or creating HIP kernels for a particular operator. MIGraphX can also target CPUs using DNNL or ZenDNN libraries.

For more information on how to install MIGraphX, refer to AMD MIGraphX Github.

#### Prerequisites

- Radeon™ Software for Linux (with ROCm) is installed

### 2.4.2 Install MIGraphX

Install MIGraphX on your computer. Once the install is completed and verified, proceed to *install Torch-MIGraphX* or *MIGraphX for ONNX Runtime*.

**To install MIGraphX**,

```
$ sudo apt install migraphx
```

Next, *verify MIGraphX installation*.

#### Verify MIGraphX installation

**To verify MIGraphX installation**,

1. Change directory to ROCm.

   ```
   cd /opt/rocm/bin
   ```

   **NOTE** This step is optional. After running apt install, the PATH variable is automatically set.

2. Verify if MIGraphX is successfully installed.

   ```
   $ ./migraphx-driver perf --model resnet50
   ```

   **Expected result**:

   ```
   will complete with no error
   ```

Next, proceed to install *Torch-MIGraphX* or *MIGraphX for ONNX Runtime* as applicable.

### 2.4.3 Install Torch-MIGraphX

Install Torch-MIGraphX using the Docker installation method, or build from source.

#### Option A: Docker installation

Using Docker provides portability, and access to a prebuilt Docker container that has been rigorously tested within AMD. Docker also cuts down compilation time, and should perform as expected without installation issues.

1. Clone the `torch_migraphx` repository.

```
git clone https://github.com/ROCmSoftwarePlatform/torch_migraphx.git
```

2. Change directory to torch-migraphx.

```
cd torch-migraphx/
```

3. Build image using the provided script.

```
sudo ./build_image.sh
```

4. Run container.

```
sudo docker run -it --network=host --device=/dev/kfd --device=/dev/dri --group-
↪add=video --ipc=host --cap-add=SYS_PTRACE --security-opt seccomp=unconfined␣
↪torch_migraphx
```

Next, *verify the Torch-MIGraphX installation*.

#### Option B: Build from source

To build from source in a custom environment, refer to the torch_migraphx repository for build steps.

Next, *verify the Torch-MIGraphX installation*.

#### Verify Torch-MIGraphX installation

Verify if the Torch-MIGraphX installation is successful.

1. Verify if torch_migraphx can be imported as a Python module.

```
python3 -c 'import torch_migraphx' 2> /dev/null && echo 'Success' || echo 'Failure
↪'
```

2. Run unit tests.

```
pytest ./torch_migraphx/tests
```

Installation is complete and the system is able to run PyTorch through the python interface library, and scripts that invoke PyTorch inference sessions.

### 2.4.4 Install MIGraphX for ONNX Runtime

Install `onnxruntime-rocm` for ROCm by following these steps.

#### Prerequisites

- *MigraphX* is installed
- Latest PyTorch ROCm release is installed

  Check Install PyTorch for ROCm for latest PIP install instructions and availability.

  **Important!** These instructions are validated for an Ubuntu 22.04 environment. Refer to the *OS support matrix* for more information.

**To install MIGraphX for ONNX Runtime**,

1. Verify that the install works correctly by performing a simple inference with MIGraphX.

```
$ migraphx-driver perf --model resnet50
```

2. Ensure that the half package is installed.

```
$ sudo apt install half
```

3. Install `onnxruntime-rocm` via Python PIP.

```
$ pip3 install onnxruntime-rocm -f https://repo.radeon.com/rocm/manylinux/rocm-
↪rel-6.0/
```

Next, *verify the MIGraphX for ONNX Runtime installation*.

#### Verify MIGraphX installation for ONNX Runtime

Verify that the install works correctly by performing a simple inference with MIGraphX.

```
$ python3
import onnxruntime as ort
ort.get_available_providers()
```

**Expected result**:

```
>>> import onnxruntime as ort
>>> ort.get_available_providers()
['MIGraphXExecutionProvider', 'ROCMExecutionProvider', 'CPUExecutionProvider']
```

This indicates that the `MIGraphXExecutionProvider` and `ROCMExecutionProvider` are now running on the system, and the proper ONNX Runtime package has been installed.

Installation is complete and ONNX Runtime is available through the Python interface library, as well as scripts that invoke ONNX Runtime inference sessions.

> For more information on the ONNX Runtime Python library, refer to Get started with ONNX Runtime in Python.

CHAPTER

THREE

# COMPATIBILITY MATRICES

This section provides information on the compatibility of ROCm™ components, Radeon™ GPUs, and the Radeon Software for Linux® version (Kernel Fusion Driver).

## 3.1 Support matrices by ROCm version

Select the applicable ROCm version for compatible OS, GPU, and framework support matrices.

### ROCm 6.0.2

#### OS support matrix

| OS | Kernel | Supported |
|---|---|---|
| Ubuntu® 22.04.3 Desktop Version with HWE | Ubuntu kernel 6.5 | Yes |

#### GPU support matrix

| ROCm Version | Radeon™ Software for Linux® Version | Supported AMD Radeon™ Hardware |
|---|---|---|
| 6.0.2 | 23.40.2 | AMD Radeon RX 7900 XTXAMD Radeon RX 7900 XTAMD Radeon RX 7900 GREAMD Radeon PRO W7900AMD Radeon PRO W7800 |

#### Framework + ROCm support matrices

View the ROCm support matrices for PyTorch and ONNX frameworks.

### PyTorch + ROCm support matrix

| PyTorch Version | ROCm Version | Comments |
|---|---|---|
| 2.1.2 | 6.0.2 | Official production support. Available from AMD.com. |
| 2.2+/Nightly | 6.0 | Available from PyTorch.org nightly builds, not tested extensively by AMD. |
| 2.2/Stable | 5.7 | Not supported for Radeon 7000 series. |

### ONNX + ROCm support matrix

| ONNX Version | ROCm Version | Comments |
|---|---|---|
| 1.17 | 6.0.2 | Official production support. Available from AMD.com. |

**Note** Refer to Installation Instructions to Get Started with ONNX Runtime for more information.

### ROCm 5.7

### OS support matrix

| OS | Kernel | Supported |
|---|---|---|
| Ubuntu® 22.04.3 Desktop Version with HWE | Ubuntu kernel 6.2 | Yes |

### GPU support matrix

| ROCm Version | Radeon™ Software for Linux® Version | Supported AMD Radeon™ Hardware |
|---|---|---|
| 5.7.0 | 23.20.00.48 | AMD Radeon RX 7900 XTXAMD Radeon RX 7900 XTAMD Radeon PRO W7900 |

### PyTorch + ROCm support matrix

| PyTorch Version | ROCm Version | Comments |
|---|---|---|
| 2.0.1 | 5.7 | Official production support. Available from AMD.com. |
| 2.1+/Nightly | 5.7 | Available from PyTorch.org nightly builds, not tested extensively by AMD. |
| 2.1/Stable | 5.6 | Not supported for Radeon 7000 series. |

## 3.2 Docker support matrix

See Docker Image Support Matrix for the latest version of the software support matrices for ROCm container releases.

CHAPTER

FOUR

# LIMITATIONS

This section provides information on software and configuration limitations.

> **Important!** Radeon™ PRO Series graphics cards are not designed nor recommended for datacenter usage. Use in a datacenter setting may adversely affect manageability, efficiency, reliability, and/or performance. GD-239.
>
> **Important!** ROCm is not officially supported on any mobile SKUs.

## 4.1 Multi-GPU configuration

Due to limited validation of ROCm™ on Radeon™ multi-GPU configuration at this time, we have identified common errors, and applicable recommendations.

> **Important!** ROCm 6.0.2 release is limited to preview support for multi-GPU configuration.
>
> At this time, only a limited amount of validation has been performed. AMD recommends only proceeding with advanced know-how and at user discretion.
>
> Visit the AI community to share feedback, and Report a bug if you find any issues.

### Recommended multi-GPU system configuration

PCIe® slots connected to the GPU must have identical PCIe lane width or bifurcation settings, and support PCIe 3.0 Atomics.

Refer to How ROCm uses PCIe Atomics for more information.

> **Example**:

✓ - GPU0 PCIe x16 connection + GPU1 PCIe x16 connection

✓ - GPU0 PCIe x8 connection + GPU1 PCIe x8 connection

**X** - GPU0 PCIe x16 connection + GPU1 PCIe x8 connection

> **Important!**
>
> - Only use PCIe slots connected by the CPU and to avoid PCIe slots connected via chipset. Refer to product-specific motherboard documentation for PCIe electrical configuration.
> - Ensure the PSU has sufficient wattage to support multiple GPUs.

### Known issues

#### Errors due to GPU and PCIe configuration

When using two AMD Radeon 7900XTX GPUs, the following HIP error is observed when running PyTorch micro-benchmarking if any one of the two GPUs are connected to a non-CPU PCIe slot (PCIe on chipset):

```
RuntimeError: HIP error: the operation cannot be performed in the present state
HIP kernel errors might be asynchronously reported at some other API call, so the␣
↪stacktrace below might be incorrect.
For debugging consider passing HIP_LAUNCH_BLOCKING=1.
Compile with `TORCH_USE_HIP_DSA` to enable device-side assertions.
```

#### Errors due to GPU and PCIe configuration with AI workloads

When using two AMD Radeon 7900XTX GPUs, both cards must be connected to a CPU-controlled PCIe slot. Not doing this might result in errors during AI workflows.

#### Potential GPU reset with some mixed graphics and compute workloads

Working with certain mixed graphics and compute workloads may result in a GPU reset on Radeon GPUs.

Currently identified scenarios include:

- Running multiple ML workloads simultaneously while using the desktop
- Running ML workloads while simultaneously using Blender/HIP

## 4.2 6.0.2 release known issues

- Running PyTorch with iGPU enabled + Discrete GPU enabled may cause crashes.
- GPU reset may occur when running multiple heavy Machine Learning workloads at same time over an extended period of time.
- Intermittent gpureset errors may be seen with Automatic 1111 webUI with IOMMU enabled. Please see https://community.amd.com/t5/knowledge-base/tkb-p/amd-rocm-tkb for suggested resolutions.
- RX 7900 GRE may exhibit a hang rather than Out Of Memory error on BERT FP32 training loads.
- Soft hang observed when running multi-queue workloads.

CHAPTER

FIVE

# AI COMMUNITY

Want to share your experiences, find answers, or contribute to resolving issues?

Explore the AMD AI Community Forum, where you will find a like-minded community, passionate about all things AI!

CHAPTER

SIX

# REPORT A BUG

Found a defect? Report issues through ROCm GitLab, and contribute to improving our user experience.