NVIDIA RTX 3050 INSTALLATION MANUAL

This is a user manual created to help with the installation of the NVIDIA RTX 3050 driver and getting the system ready for configuring Deep Learning Applications.

Installing GeForce RTX 3050 NVIDIA Driver

Q. Steps to install the GeForce RTX 3050 NVIDIA driver on the computer.

Here are the steps to install the NVIDIA GeForce RTX 3050 graphics card driver on a computer:

1. Go to the NVIDIA website (www.nvidia.com) and click on the "Drivers" section.
2. Select "GeForce" from the product type drop-down menu and "GeForce RTX 3050" from the product series drop-down menu.
3. Choose your operating system (e.g. Windows 10) and click "Search."
4. Download the latest available driver for the GeForce RTX 3050 by clicking the "Download" button.
5. Save the driver file to your computer and double-click the file to start the installation process.
6. Follow the on-screen instructions to install the driver.
7. Once the installation is complete, restart your computer.
8. Verify that the NVIDIA GeForce RTX 3050 is functioning properly by checking the Device Manager or running benchmarks.

Note: If you already have an older version of the NVIDIA driver installed, you may need to uninstall it first before installing the new driver. You can do this using the "Programs and Features" or "Apps & Features" control panel in Windows.

Q. How to check the driver version of NVIDIA Graphic card

Here are the steps to check the version of the NVIDIA graphics card driver on a Windows computer:

1. Right-click the Windows Start button and select "Device Manager."

Prepared by: VANI SETH 201B299

2. Expand the "Display adapters" section.
3. Right-click on your NVIDIA graphics card and select "Properties."
4. Go to the "Driver" tab.
5. The version of the driver is listed under "Driver version."

Alternatively, you can also check the version of the NVIDIA graphics card driver by using the NVIDIA Control Panel:

1. Right-click the Windows desktop and select "NVIDIA Control Panel."
2. Go to the "System Information" section.
3. The version of the NVIDIA graphics card driver is listed under "Driver Version."

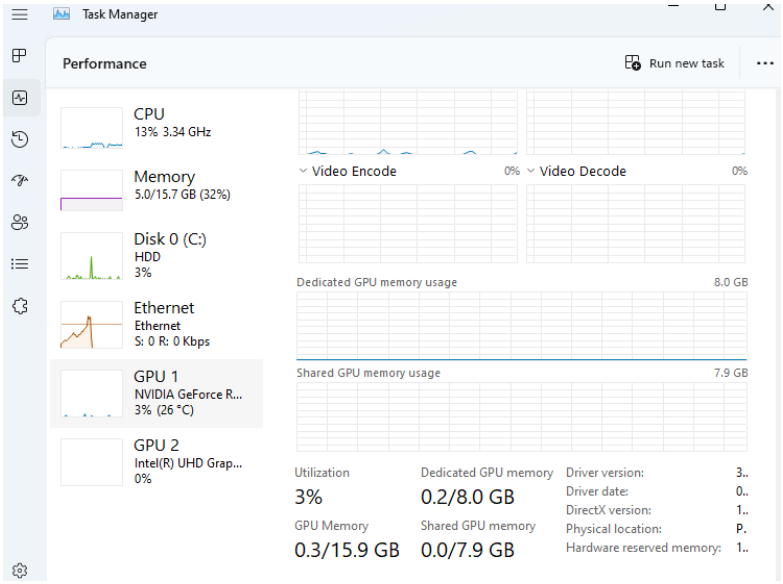
Note: The specific steps to check the driver version may vary depending on the operating system and version you are using.

Q. What download type to choose in NVIDIA Driver downloads GRD or SD

In NVIDIA driver downloads, you have the option to choose between "GRD (Graphics Driver)" and "SD (System Driver)".

1. GRD (Graphics Driver): This type of driver is specifically designed to enhance the performance and functionality of your NVIDIA graphics card. It includes support for all the latest features and optimizations to ensure the best performance for your graphics card.
2. SD (System Driver): This type of driver is designed to enhance the performance and stability of your entire system, including your NVIDIA graphics card. It includes support for features like NVIDIA GeForce Experience and other system-level optimizations.

Typically, the GRD (Graphics Driver) is recommended for most users as it provides the best performance and support for your NVIDIA graphics card. However, the SD (System Driver) is also a good option if you require enhanced system-level performance and stability. The choice between the two will depend on your specific needs and requirements.



Once the installation of the driver has been completed, on opening the Task Manager you can check another GPU has been added named as NVIDIA GeForce RTX 3050.

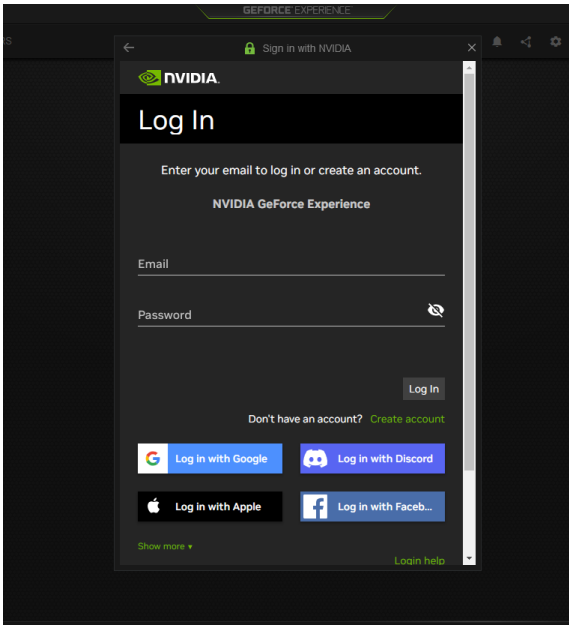
Q. Which installation option to choose in the NVIDIA installer?

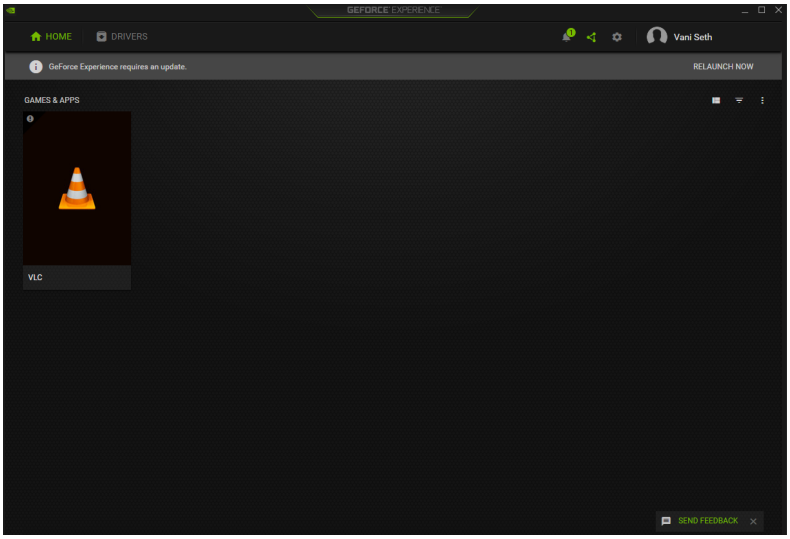
When installing NVIDIA graphics card drivers using the NVIDIA installer, you will typically have two options to choose from: "Custom" and "Express".

1. Custom: This option allows you to customize the installation process and select the specific components you want to install. You can choose to install only the graphics driver, only the NVIDIA Control Panel, or both. This option is recommended for advanced users who want to control the installation process and have specific requirements.
2. Express: This option is the recommended option for most users. It installs the graphics driver and the NVIDIA Control Panel, along with any necessary system components. This option is quick, easy, and provides the best overall experience for most users.

If you are unsure which option to choose, it is recommended to choose the Express option as it provides a comprehensive installation and the best overall experience for most users.

Prepared by: VANI SETH 201B299





Make an account if you do not already have or login with Google. Once you are logged in, your dashboard will open.

Once the dashboard opens, you have successfully installed NVIDIA Driver in your system.

Now we need to make our system capable of running and configuring deep learning programs.

Configuring Deep Learning Programs

To configure Deep Learning Programs while using the GPU driver we can do this in two ways:

1. Installing all the toolkits required by the NVIDIA GPU Driver.
2. Installing Anaconda and using NVIDIA GPU Driver through that

Prepared by: VANI SETH 201B299

MANUAL INSTALLATION OF ALL THE DEPENDENCIES.

Q. How to configure deep learning programs using GForce RTX 3050 NVIDIA Driver

The configuration process for deep learning programs using the NVIDIA GeForce RTX 3050 driver will depend on the specific deep learning software you are using. However, here are some general steps you can follow to configure deep learning programs:

1. Install the latest NVIDIA graphics card driver for your GeForce RTX 3050. You can download the driver from the NVIDIA website (www.nvidia.com) and follow the instructions provided in the installation wizard.
2. Install the deep learning software you want to use. This can be a deep learning framework such as TensorFlow, PyTorch, or Caffe, or a deep learning application such as NVIDIA DIGITS or CUDA-Z.
3. Verify that the deep learning software you installed is using the NVIDIA GPU. You can do this by checking the system requirements for the specific software or by using system monitoring tools like NVIDIA System Management Interface (nvidia-smi) or GPU-Z.
4. Configure the deep learning software for your specific use case. This may include setting the number of GPUs to use, specifying the GPU memory to allocate, and configuring other performance-related settings.
5. Start using the deep learning software to build and train your deep learning models.

Note: The exact steps for configuring deep learning programs may vary depending on the specific deep learning software you are using. Be sure to consult the documentation and support resources provided by the software vendor for more information.

Q. How to install TensorFlow on GeForce Workspace

To install TensorFlow on the GeForce Workspace, you need to follow these steps:

1. Install the latest version of Python. TensorFlow supports Python 3.5 or higher. You can download the latest version of Python from the official Python website (python.org) and follow the installation instructions.

Prepared by: VANI SETH 201B299

2. Install the NVIDIA GPU drivers for your GeForce RTX 3050 graphics card. You can download the latest drivers from the NVIDIA website (www.nvidia.com) and follow the instructions provided in the installation wizard.
3. Install the NVIDIA CUDA Toolkit. TensorFlow requires the NVIDIA CUDA Toolkit to use your GPU. You can download the latest version of the CUDA Toolkit from the NVIDIA website (developer.nvidia.com/cuda-downloads) and follow the installation instructions.
4. Install the NVIDIA cuDNN library. TensorFlow also requires the NVIDIA cuDNN library for GPU acceleration. You can download the latest version of cuDNN from the NVIDIA website (developer.nvidia.com/cudnn) and follow the installation instructions.
5. Install TensorFlow. You can install TensorFlow using the pip package manager by running the following command in your terminal or command prompt:

```
pip install tensorflow-gpu
```

6. Verify the installation. To verify the installation, you can start a Python session and run the following command:

```
import tensorflow as tf;
print(tf.reduce_sum(tf.random.normal([1000, 1000])))
```

   This will perform a simple computation using TensorFlow and verify that it is working correctly.

These are the basic steps to install TensorFlow on the GeForce Workspace. For more information on TensorFlow, including system requirements and usage instructions, refer to the TensorFlow website (tensorflow.org).

Q. How to install CUDA toolkit

To install the NVIDIA CUDA Toolkit on your computer, you need to follow these steps:

1. Download the CUDA Toolkit installer. You can download the latest version of the CUDA Toolkit from the NVIDIA website (developer.nvidia.com/cuda-downloads).

2. Run the CUDA Toolkit installer. Double-click the downloaded installer file and follow the on-screen instructions to start the installation process.
3. Accept the license agreement. Read the license agreement and accept it to continue with the installation.
4. Choose the installation type. Select "Custom (Advanced)" installation type to choose which components of the CUDA Toolkit to install. You can choose to install only the necessary components or install all components, depending on your needs.
5. Choose the installation location. Specify the location where you want to install the CUDA Toolkit. The default location is usually fine.
6. Install the CUDA Toolkit. Click the "Install" button to start the installation process. Wait for the installation to complete.
7. Update your PATH environment variable. Add the CUDA Toolkit bin directory to your PATH environment variable. You can do this by adding the following line to your shell startup script (e.g., .bashrc or .bash_profile):

```
export PATH=/usr/local/cuda/bin:$PATH
```

8. Update your LD_LIBRARY_PATH environment variable. Add the CUDA Toolkit lib64 directory to your LD_LIBRARY_PATH environment variable. You can do this by adding the following line to your shell startup script (e.g., .bashrc or .bash_profile):\

```
export LD_LIBRARY_PATH=/usr/local/cuda/lib64:$LD_LIBRARY_PATH
```

9. Verify the installation. To verify the installation, you can run the following command in the terminal or command prompt:

```
nvidia-smi
```

This will display information about your NVIDIA GPU, including the CUDA Toolkit version.

These are the basic steps to install the NVIDIA CUDA Toolkit on your computer. For more information on the CUDA Toolkit, including system requirements and usage instructions, refer to the NVIDIA CUDA Toolkit documentation on the NVIDIA website.

Q. which installer type to use while downloading CUDA toolkit?

When downloading the NVIDIA CUDA Toolkit, you have two types of installers to choose from:

1. Network Installer: This is a smaller installer that downloads the necessary components from the internet during installation. This is a good option if you have a fast internet connection and limited disk space.
2. Local Installer: This is a larger installer that includes all the necessary components for installation. This is a good option if you have a slow or unreliable internet connection or if you want to save time by not having to download components during installation.

In general, it's recommended to use the Local Installer if you have a slow or unreliable internet connection or if you want to save time. If you have a fast internet connection and limited disk space, you can use the Network Installer. The choice between the two types of installers ultimately depends on your specific requirements and constraints.

```
PS C:\Users\201b299> nvcc --version
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2023 NVIDIA Corporation
Built on Fri_Jan__6_19:04:39_Pacific_Standard_Time_2023
Cuda compilation tools, release 12.0, V12.0.140
Build cuda_12.0.r12.0/compiler.32267302_0
PS C:\Users\201b299>
```

Checking the version to verify the installation.

Q. How to install cuDNN to NVIDIA toolkit

Here is a general outline of the steps to install the NVIDIA CUDA Deep Neural Network Library (cuDNN):

1. Download cuDNN: Go to the NVIDIA website (https://developer.nvidia.com/cudnn) and log in or create an account if necessary. Download the latest version of cuDNN that is compatible with your version of the CUDA Toolkit.
2. Unpack the cuDNN archive: After downloading the cuDNN archive, unpack it to a directory of your choice. For example, you can unpack it to the ` **~/Downloads**` directory by running the following command in a terminal:

```
tar -xzvf cudnn-<version>-<platform>.tgz -C ~/Downloads
```

   Replace **<version>** with the version of cuDNN you downloaded, and **<platform>** with the platform you are using (for example, **linux-x64** for a 64-bit Linux system).

3. Copy the cuDNN libraries: Copy the cuDNN libraries to the CUDA Toolkit library directory. For example, if you have installed the CUDA Toolkit in the default location, you can run the following commands in a terminal:

```
sudo cp ~/Downloads/cuda/include/cudnn.h /usr/local/cuda/include/
sudo cp ~/Downloads/cuda/lib64/libcudnn* /usr/local/cuda/lib64/
sudo chmod a+r /usr/local/cuda/include/cudnn.h
/usr/local/cuda/lib64/libcudnn*
```

4. Update environment variables: Update the environment variables to include the path to the CUDA Toolkit bin and lib64 directories. This can be done by editing the shell startup script as described in the previous answer.
5. Verify the installation: Verify that the cuDNN library is installed correctly by running the following command in a terminal:

```
cat /usr/local/cuda/include/cudnn.h | grep CUDNN_MAJOR -A 2
```

You should see the cuDNN version and API version information in the output.

Prepared by: VANI SETH 201B299

These are the general steps to install the NVIDIA CUDA Deep Neural Network Library (cuDNN). For more information, refer to the cuDNN documentation on the NVIDIA website

Q. how to install tensorflow-gpu on NVIDIA

You can install the TensorFlow GPU version by using pip. The steps are as follows:

1. Make sure you have the NVIDIA CUDA Toolkit and cuDNN installed on your system.
2. Open a terminal or command prompt window.
3. Run the following command to install TensorFlow GPU version:

```
pip install tensorflow-gpu
```

Note: Make sure to use the appropriate pip command, depending on the environment you are using. If you are using a virtual environment, activate it before installing TensorFlow.

After the installation is complete, you can verify that TensorFlow GPU is installed by running the following code in Python:

```
import tensorflow as tf
print("TensorFlow version:", tf.__version__)
print("GPU is", "available" if tf.test.is_gpu_available() else "not
available")
```

NOTE: If you are having trouble installing tensorflow via this process, use method 2 to install tensorflow and use the NVIDIA driver.

INSTALLING ANACONDA TO USE NVIDIA GPU DRIVER

Q. How to install tensorflow if you are having trouble installing it from the terminal?

You can install TensorFlow using the Anaconda distribution by following these steps:

1. Open the Anaconda Prompt.
2. Create a new environment with the following command:

```
conda create -n env_name
```

3. Replace "env_name" with the desired name of your environment.
4. Activate the new environment with the following command:

```
conda activate env_name
```

5. Install TensorFlow GPU version with the following command:

```
conda install tensorflow-gpu
```

Note: Make sure that your system has the NVIDIA CUDA Toolkit and cuDNN installed before installing TensorFlow GPU.

After the installation is complete, you can verify that TensorFlow GPU is installed by opening the Anaconda prompt and activating the environment, then running the following code in Python:

```
import tensorflow as tf
print("TensorFlow version:", tf.__version__)
print("GPU is", "available" if tf.test.is_gpu_available() else "not
available")
```

Prepared by: VANI SETH 201B299

Q. how to install anaconda on windows

You can install Anaconda on Windows by following these steps:

1. Download the Anaconda installation package for Windows from the official Anaconda website (https://www.anaconda.com/products/distribution).
2. Open the installation package by double-clicking on it.
3. Follow the steps in the installation wizard to install Anaconda. You can accept the default options, or you can customize the installation options as needed.
4. Once the installation is complete, open the Anaconda Navigator by searching for "Anaconda Navigator" in the Windows search bar.
5. You can now use Anaconda to create and manage environments, install packages, and run Jupyter notebooks.

Note: To run the Anaconda command line interface (Anaconda Prompt), you will need to open the Windows Start menu and search for "Anaconda Prompt."

Q. how to install tensorflow-gpu using anaconda

To install TensorFlow-GPU using Anaconda, follow these steps:

1. Open Anaconda Prompt: Follow the steps mentioned in the previous answer to open the Anaconda Prompt terminal.
2. Create a virtual environment: You can create a virtual environment to keep the TensorFlow installation separate from other packages on your system. To create a virtual environment, run the following command in the Anaconda Prompt:

```
conda create --name tensorflow-gpu python=3.7
```

3. Activate the virtual environment: To activate the virtual environment, run the following command:

```
conda activate tensorflow-gpu
```

Prepared by: VANI SETH 201B299

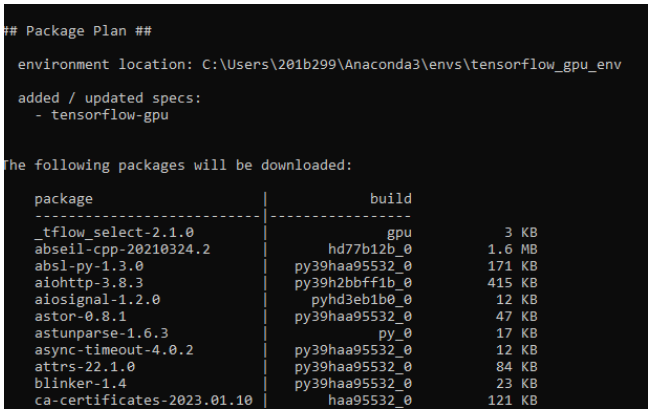
4. Install TensorFlow-GPU: To install TensorFlow-GPU, run the following command:

```
conda install tensorflow-gpu
```

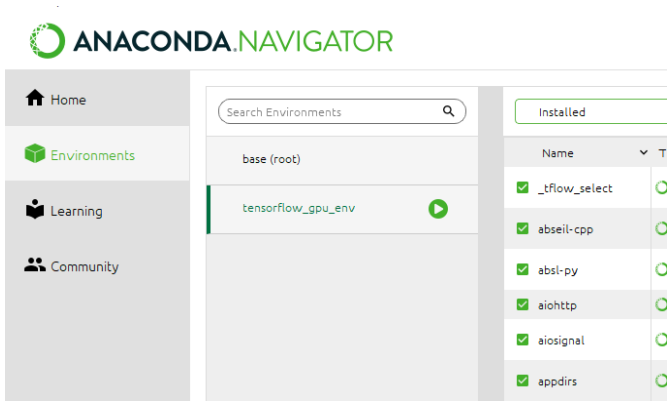
5. Verify the installation: To verify that TensorFlow-GPU has been installed, run the following command:

```
python -c "import tensorflow as tf;
print(tf.reduce_sum(tf.random.normal([1000, 1000])))"
```

If TensorFlow-GPU has been installed correctly, you should see the output of the reduce_sum operation, and you should not encounter any errors or warnings.



Making environment in anaconda prompt



Environment made successfully

```
(tensorflow_gpu_env) C:\>python -c "import tensorflow as tf; print(tf.reduce_sum(tf.random.normal([1000, 1000])))"
2023-02-11 10:22:02.189345: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow binary is optimized wi
ctions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2023-02-11 10:22:02.586050: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1510] Created device /job:localhost/repl
orce RTX 3050, pci bus id: 0000:01:00.0, compute capability: 8.6
tf.Tensor(178.11017, shape=(), dtype=float32)
```

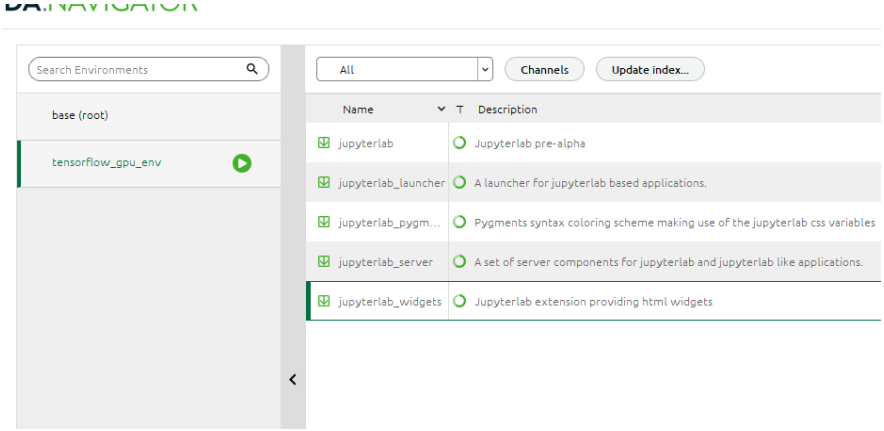
Verifying if the installation has been successful

Q. how to install jupyter notebook in anaconda

You can install Jupyter Notebook in Anaconda by following these steps:

1. Open the Anaconda Navigator.
2. Click on the "Environments" tab in the left sidebar.
3. Click on the triangle next to the environment where you want to install the Jupyter Notebook.
4. Click on "Open Terminal" to open a terminal window for that environment.
5. In the terminal window, type **conda install jupyter** and press Enter.
6. Anaconda will download and install Jupyter Notebook along with any required dependencies.

After the installation is complete, you can launch Jupyter Notebook from the Anaconda Navigator by clicking on the "Home" tab in the left sidebar, and then clicking the "Launch" button next to the Jupyter Notebook icon.



Installing all the packages of Jupyter Notebook in the environment

Prepared by: VANI SETH 201B299

Q. Anaconda terminal command to check the number of gpu

You can use the following Anaconda terminal command to check the number of GPUs:

```
nvidia-smi -L
```

This will display a list of the available GPUs on your system along with their corresponding IDs.

Q. anaconda command to check which gpu is working

To check which GPU is working in Anaconda, you can use the following command in the Anaconda prompt:

```
python -c "import tensorflow as tf;
print(tf.config.list_physical_devices('GPU'))"
```

This will output a list of physical GPUs available on the machine, and which one is being used by TensorFlow.

```
(tensorflow_gpu_env) C:\Users\201b299>nvidia-smi -L
GPU 0: NVIDIA GeForce RTX 3050 (UUID: GPU-abfbd580-2804-d71b-ec80-c5745c8ab3c4)

(tensorflow_gpu_env) C:\Users\201b299>python -c "import tensorflow as tf; print(tf.config.list_physical_devices('GPU'))"

[PhysicalDevice(name='/physical_device:GPU:0', device_type='GPU')]

(tensorflow_gpu_env) C:\Users\201b299>
```

Q. Steps to run a simple deep learning program using gpu drivers

Here are the general steps to run a simple deep learning program using GPU drivers:

1. Install the necessary GPU drivers, such as the NVIDIA driver and CUDA toolkit.
2. Install a deep learning framework that supports GPU, such as TensorFlow or PyTorch.
3. Write a simple deep learning program in your preferred programming language, such as Python.
4. Ensure that your program is set up to use the GPU for training and inference.

Prepared by: VANI SETH 201B299

5. Run the program and verify that it is using the GPU by monitoring the GPU usage and performance.
6. Adjust the program as necessary to improve performance, such as by adjusting the batch size, optimizing the model architecture, or using mixed-precision training.

Here is an example of a simple TensorFlow program in Python that trains a small neural network on the MNIST dataset using the GPU:

```
import tensorflow as tf
from tensorflow.keras.datasets import mnist

# Load the MNIST dataset
(x_train, y_train), (x_test, y_test) = mnist.load_data()

# Preprocess the data
x_train = x_train.reshape((60000, 28, 28, 1)) / 255.0
x_test = x_test.reshape((10000, 28, 28, 1)) / 255.0

# Define the model architecture
model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu',
input_shape=(28, 28, 1)),
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(10, activation='softmax')
])

# Compile the model
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

# Train the model on the GPU
with tf.device('/GPU:0'):
    model.fit(x_train, y_train, epochs=5, validation_data=(x_test,
y_test))
```

In this example, the with **tf.device('/GPU:0'):** block specifies that the training should be done on the first available GPU. You can also use other devices, such as **/CPU:0** or **/device:GPU:1**, depending on your system configuration.

```
Epoch 1/5
1875/1875 [==============================] - 60s 2ms/step - loss: 0.2179 - accu
racy: 0.9380 - val_loss: 0.0937 - val_accuracy: 0.9719
Epoch 2/5
1875/1875 [==============================] - 3s 2ms/step - loss: 0.0846 - accur
acy: 0.9756 - val_loss: 0.0696 - val_accuracy: 0.9791
Epoch 3/5
1546/1875 [=======================>......] - ETA: 0s - loss: 0.0640 - accuracy:
0.9808
```

All the epochs are running, this program has compiled successfully.