role: you are an expert Azure cloud, mainly in Azure networking and solution architect. task: If I create an Azure VM with Ubuntu Server 22.04 LTS - x64 Gen2 OS + Standard\_NV36ads\_A10\_v5 - 36 vcpus, 440 GB memory + 1 TB Premium SSD disk size. Could you please tell me the version of GPU Card used in this azure VM and the detailed steps to install the compatible drivers for this GPU Card.

**Copilot said: Absolutely! Here’s a detailed answer based on**

Absolutely! Here’s a detailed answer based on your Azure VM configuration:

**1. GPU Card in Standard\_NV36ads\_A10\_v5**

The **Standard\_NV36ads\_A10\_v5** Azure VM uses the **NVIDIA A10** GPU.

* Each VM in this series is backed by **NVIDIA A10 Tensor Core GPUs**.
* The A10 is based on NVIDIA’s Ampere architecture and is designed for both graphics and compute workloads.

**2. Steps to Install the Compatible NVIDIA Drivers (Ubuntu 22.04 LTS)**

Follow these steps on your Ubuntu Server 22.04 LTS VM to install the GPU drivers and CUDA toolkit:

**A. Update and Prepare the System**

bash

sudo apt update

sudo apt upgrade -y

sudo apt install -y build-essential dkms

**B. Disable Nouveau Driver (if loaded)**

bash

sudo bash -c "echo 'blacklist nouveau' > /etc/modprobe.d/blacklist-nouveau.conf"

sudo bash -c "echo 'options nouveau modeset=0' >> /etc/modprobe.d/blacklist-nouveau.conf"

sudo update-initramfs -u

sudo reboot

**Note:** Reboot is necessary to unload Nouveau if it was loaded.

**C. Install the NVIDIA Drivers**

**Option 1: Using Ubuntu’s Official Repositories (Recommended)**

bash

sudo apt install -y ubuntu-drivers-common

sudo ubuntu-drivers autoinstall

sudo reboot

* This will automatically detect and install the recommended NVIDIA driver for the A10 (typically version 550+ for A10 GPUs).

**Option 2: Download the Latest Driver from NVIDIA (for full feature set)**

1. Download the [latest driver for NVIDIA A10 from NVIDIA’s official site](https://www.nvidia.com/Download/driverResults.aspx/).
2. Make the file executable and install:

bash

chmod +x NVIDIA-Linux-x86\_64-\*.run

sudo ./NVIDIA-Linux-x86\_64-\*.run

1. Follow the prompts in the installer.

**D. Install CUDA Toolkit (Optional, for compute workloads)**

1. Download the [CUDA Toolkit for Ubuntu 22.04](https://developer.nvidia.com/cuda-downloads).
2. Follow the instructions provided for Ubuntu 22.04 (choose network or local installer).

Example (network installer):

bash

wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2204/x86\_64/cuda-ubuntu2204.pin

sudo mv cuda-ubuntu2204.pin /etc/apt/preferences.d/cuda-repository-pin-600

wget https://developer.download.nvidia.com/compute/cuda/12.5.0/local\_installers/cuda-repo-ubuntu2204-12-5-local\_12.5.0-1\_amd64.deb

sudo dpkg -i cuda-repo-ubuntu2204-12-5-local\_12.5.0-1\_amd64.deb

sudo cp /var/cuda-repo-ubuntu2204-12-5-local/cuda-\*-keyring.gpg /usr/share/keyrings/

sudo apt-get update

sudo apt-get -y install cuda

* Adjust the version as needed for your requirements.

**E. Verify Installation**

bash

nvidia-smi

* This command should show the NVIDIA A10 GPU and the installed driver version.

**Summary Table**

| **Azure VM Size** | **GPU Model** | **Driver Needed** | **Notes** |
| --- | --- | --- | --- |
| Standard\_NV36ads\_A10\_v5 | NVIDIA A10 | NVIDIA Linux Driver 550+ | Ubuntu 22.04 supported, CUDA optional |

**References:**

* [Azure NVads A10 v5 Documentation (Microsoft)](https://learn.microsoft.com/en-us/azure/virtual-machines/nvads-a10-v5-series)
* [NVIDIA A10 Product Page](https://www.nvidia.com/en-us/data-center/a10/)
* [NVIDIA Driver Downloads](https://www.nvidia.com/download/index.aspx)

If you need step-by-step screenshots or want to automate this process with cloud-init or scripts, let me know!

**You said: Yes, I would like to have this**

Yes, I would like to have this process automated so that I just need to run a script or command after conneting to this VM using SSH and all these GPU drivers and CUDA toolkit gets installed in this VM, so that I can start using it straight away

**Copilot said: Great! Here’s a ready-to-use Bash script that**

Great! Here’s a ready-to-use Bash script that automates the installation of the NVIDIA A10 GPU drivers and the CUDA Toolkit on Ubuntu 22.04 LTS.  
You just need to SSH into your Azure VM and run this script as root (or with sudo).  
The script covers: system update, prerequisite installation, nouveau driver blacklisting, NVIDIA driver installation, and CUDA Toolkit installation.

**install\_nvidia\_a10\_cuda.sh**

#!/bin/bash

set -e

# 1. Update system and install required packages

sudo apt update && sudo apt upgrade -y

sudo apt install -y build-essential dkms wget curl linux-headers-$(uname -r)

**Instructions to Use**

1. **SSH into your VM:**

sh

ssh <your-vm-user>@<your-vm-ip>

1. **Copy and run the script:**

sh

wget https://raw.githubusercontent.com/<your-repo>/install\_nvidia\_a10\_cuda.sh

chmod +x install\_nvidia\_a10\_cuda.sh

./install\_nvidia\_a10\_cuda.sh

Or, paste the script above into a file called install\_nvidia\_a10\_cuda.sh and run it.

1. **Follow the prompts:**  
   The script will ask to reboot after blacklisting the Nouveau driver and again after the NVIDIA driver installation.  
   After each reboot, you need to re-run the script for the next steps.

**NOTES**

* This script is idempotent for each step, but you must re-run it after each reboot until the process completes.
* The CUDA version (12.5.0 as shown) can be updated to the latest if needed.
* After completion, you can verify your GPU and driver with nvidia-smi and CUDA with nvcc --version.

Let me know if you want a more seamless (single-run) version, or if you want a cloud-init or Azure Custom Script Extension version for full automation at VM deployment time!

you said "After each reboot, you need to re-run the script for the next steps." so in each re-run which command I should use to re-run the script

**Copilot said: Great question! After each reboot, you \*\*do**

Great question! After each reboot, you **do not need to copy or recreate the script**—you just need to re-run the same script file you created earlier.

**If you named your script install\_nvidia\_a10\_cuda.sh, then after each reboot, simply run:**

bash

./install\_nvidia\_a10\_cuda.sh

or, if you need to run it with sudo:

bash

sudo ./install\_nvidia\_a10\_cuda.sh

**Tip:**

* Make sure you’re in the same directory as the script file. If not, navigate to that directory using cd /path/to/your/script/.
* If you get a “permission denied” error, use chmod +x install\_nvidia\_a10\_cuda.sh to make it executable.

**Why this is necessary**

The script includes steps that require a reboot (like disabling the Nouveau driver and installing the NVIDIA driver). After each reboot, re-running the script will continue from where it left off, skipping already-completed steps.

If you want an improved, single-run script that automatically resumes after reboot using systemd or a flag file, let me know!

after executing the script install\_nvidia\_a10\_cuda.sh, how can I confirm that the GPU Drivers and CUDA toolkits are installed properly?

**Copilot said: To confirm that the NVIDIA GPU drivers**

To confirm that the NVIDIA GPU drivers and CUDA toolkit are installed properly on your Azure Ubuntu VM, follow these steps after running your install\_nvidia\_a10\_cuda.sh script:

**1. Check NVIDIA Driver Installation**

Run:

bash

nvidia-smi

**Expected Output:**

* You should see a table listing your NVIDIA A10 GPU.
* The driver version should be shown (e.g., 550.xx or similar).
* GPU utilization and other details will be displayed.

**Example:**

Code

+-----------------------------------------------------------------------------+

| NVIDIA-SMI 550.54.14 Driver Version: 550.54.14 CUDA Version: 12.5 |

|-------------------------------+----------------------+----------------------+

| GPU Name Persistence-M| Bus-Id Disp.A | Volatile Uncorr. ECC |

| Fan Temp Perf Pwr:Usage/Cap| Memory-Usage | GPU-Util Compute M. |

|===============================+======================+======================|

| 0 NVIDIA A10 On | 00000000:00:1E.0 Off | 0 |

| 30% 28C P8 16W / 150W | 0MiB / 23040MiB | 0% Default |

+-----------------------------------------------------------------------------+

If you see this, your NVIDIA driver is working.

**2. Check CUDA Toolkit Installation**

**A. Verify CUDA Compiler**

bash

nvcc --version

**Expected Output:**

* Displays CUDA compiler version (e.g., release 12.5).

**Example:**

Code

nvcc: NVIDIA (R) Cuda compiler driver

Copyright (c) 2005-2024 NVIDIA Corporation

Built on Fri\_May\_10\_2024

Cuda compilation tools, release 12.5, V12.5.52

**B. Check CUDA Toolkit Path**

bash

echo $PATH | grep cuda

echo $LD\_LIBRARY\_PATH | grep cuda

* Make sure /usr/local/cuda/bin and /usr/local/cuda/lib64 are present.

**3. Run a CUDA Sample (Optional, for full confirmation)**

**A. Copy and build CUDA samples:**

bash

cp -r /usr/local/cuda/samples ~/cuda\_samples

cd ~/cuda\_samples/1\_Utilities/deviceQuery

make

./deviceQuery

**B. Expected Output:**

* At the end, it should say Result = PASS and show detailed info for your NVIDIA A10 GPU.

**Summary Table**

| **Command** | **What it checks** | **Expected Output** |
| --- | --- | --- |
| nvidia-smi | NVIDIA driver & GPU detection | Table with NVIDIA A10 and driver info |
| nvcc --version | CUDA compiler presence/version | CUDA version info |
| ./deviceQuery | CUDA runtime & GPU functionality | GPU info and Result = PASS |

**If any of these commands fail:**

* The drivers or toolkit may not be installed or configured correctly.
* Review error messages and re-run the install script as needed.

Let me know if you get stuck, or paste the output here for troubleshooting!

okay. in that case could you please create a script to install Crest 3.0.2, Xtb 6.5.1, gromacs 2023.3

**Copilot said: ```bash name=install\_crest\_xtb\_gromacs.sh #!/bin/bash set -e # Define base installation directory INSTALL\_DIR="$HOME/chemsoft" mkdir**

**install\_crest\_xtb\_gromacs.sh**

#!/bin/bash

set -e

# Define base installation directory

INSTALL\_DIR="$HOME/chemsoft"

mkdir -p "$INSTALL\_DIR"

**Instructions:**

1. Save this script as install\_crest\_xtb\_gromacs.sh.
2. Run:

bash

chmod +x install\_crest\_xtb\_gromacs.sh

./install\_crest\_xtb\_gromacs.sh

1. After installation completes, run:

bash

source ~/.bashrc

1. Now, crest, xtb, and gmx should be available in your shell.

after executing the script install\_crest\_xtb\_gromacs.sh, how can I confirm that the both CREST, XTB and Gromacs are installed properly?

**Copilot said: After running your `install\_crest\_xtb\_gromacs.sh` script, you can**

After running your install\_crest\_xtb\_gromacs.sh script, you can confirm that **CREST**, **xtb**, and **GROMACS** are installed properly with the following checks:

**1. Confirm CREST Installation**

bash

crest --help

**Expected output:**  
A help message describing CREST’s usage and options.

**2. Confirm xtb Installation**

bash

xtb --help

**Expected output:**  
A help message for xtb, showing available commands and options.

**3. Confirm GROMACS Installation**

**A. Make sure GROMACS is in your environment**

If you have not opened a new terminal/session after installation, run:

bash

source ~/.bashrc

or manually source the GROMACS environment:

bash

source ~/chemsoft/gromacs-2023.3/bin/GMXRC

**B. Check GROMACS version**

bash

gmx --version

**Expected output:**  
Should display GROMACS version **2023.3** with build information.

**4. (Optional) Quick Functionality Test**

You can also run:

bash

crest

xtb

gmx help

Each should show a help message or available subcommands (not "command not found").

**5. Troubleshooting**

If any command returns "command not found":

* Make sure you ran source ~/.bashrc (to update your PATH).
* Check if the binaries exist in ~/chemsoft/crest-3.0.2/, ~/chemsoft/xtb-6.5.1/bin/, and ~/chemsoft/gromacs-2023.3/bin/.

**Summary Table**

| **Software** | **Command** | **Expected Output** |
| --- | --- | --- |
| CREST | crest --help | CREST help/usage message |
| xtb | xtb --help | xtb help/usage message |
| GROMACS | gmx --version | GROMACS version info (2023.3) |
| GROMACS | gmx help | List of GROMACS tools/commands |

If all commands produce help or version info, your installation is successful!

now could you please create a script to install and configure ORCA 5.0.4 and Gaussian 16 Rev.B.01 (linux64) in this Azure VM

**Copilot said: ```bash name=install\_orca\_gaussian.sh #!/bin/bash set -e # Set installation directory INSTALL\_DIR="$HOME/chemsoft" mkdir -p**

**install\_orca\_gaussian.sh**v2

#!/bin/bash

set -e

# Set installation directory

INSTALL\_DIR="$HOME/chemsoft"

mkdir -p "$INSTALL\_DIR"

**How to use:**

1. **Before running:**
   * Download and place the following files in $HOME/chemsoft:
     + orca\_5\_0\_4\_linux\_x86-64\_shared\_openmpi411.tar.xz (from [ORCA Forum](https://orcaforum.kofo.mpg.de/app.php/dlext/), login required)
     + g16\_B.01\_linux64.tar (from your licensed Gaussian distribution)
2. **Run the script:**

bash

chmod +x install\_orca\_gaussian.sh

./install\_orca\_gaussian.sh

1. **After install:**
   * Run source ~/.bashrc
   * For **Gaussian**, follow the vendor's instructions to set up your license/host files as needed.
   * Both programs should now be available in your shell.

**Let me know if you need cluster configuration, scheduler integration, or automated license file setup!**

could you please detail the steps to download orca\_5\_0\_4\_linux\_x86-64\_shared\_openmpi411.tar.xz and g16\_B.01\_linux64.tar from their location and place them in the folder $HOME/chemsoft in this Azure Ubuntu VM by connecting to it using SSH

**Copilot said: Certainly! Here are the step-by-step instructions to**

Certainly! Here are the step-by-step instructions to download **ORCA 5.0.4** and **Gaussian 16 Rev.B.01 (linux64)** installers and place them in $HOME/chemsoft on your Azure Ubuntu VM via SSH:

**1. Prepare Your Azure Ubuntu VM**

**a. Connect to your VM via SSH:**

bash

ssh <your-username>@<your-vm-public-ip>

**b. Create the target directory:**

bash

mkdir -p $HOME/chemsoft

cd $HOME/chemsoft

**2. Download ORCA 5.0.4**

**ORCA is only available to registered academic users via the official ORCA forum.**

**a. On Your Local Machine (with browser):**

1. Go to the [ORCA Forum Download Page](https://orcaforum.kofo.mpg.de/app.php/dlext/?cat=3). <https://orcaforum.kofo.mpg.de/app.php/dlext/?cat=3>
2. **Log in** with your registered account (registration may require proof of academic affiliation).
3. Locate and download:  
   orca\_5\_0\_4\_linux\_x86-64\_shared\_openmpi411.tar.xz  
   Save this file to your local computer.

**b. Upload the ORCA installer to your Azure VM:** From your local terminal (replace <your-username> and <your-vm-public-ip>):

bash

scp /path/to/orca\_5\_0\_4\_linux\_x86-64\_shared\_openmpi411.tar.xz <your-username>@<your-vm-public-ip>:~/chemsoft/

or, if you are already in the chemsoft directory on your VM, you can use:

bash

scp /path/to/orca\_5\_0\_4\_linux\_x86-64\_shared\_openmpi411.tar.xz <your-username>@<your-vm-public-ip>:~/chemsoft/

**3. Download Gaussian 16 Rev.B.01 (linux64)**

**Gaussian can only be obtained by licensed users, typically via your institution or vendor.**

**a. On Your Local Machine (with browser):**

1. Log in to your institution’s Gaussian distribution site or contact your Gaussian software administrator.
2. Download the file:  
   g16\_B.01\_linux64.tar  
   Save it to your local computer.

**b. Upload the Gaussian installer to your Azure VM:** From your local terminal:

bash

scp /path/to/g16\_B.01\_linux64.tar <your-username>@<your-vm-public-ip>:~/chemsoft/

**4. Verify on the Azure VM**

After transferring both files, SSH into your VM and run:

bash

ls -lh $HOME/chemsoft

You should see:

Code

orca\_5\_0\_4\_linux\_x86-64\_shared\_openmpi411.tar.xz

g16\_B.01\_linux64.tar

**Summary Table**

| **Software** | **Download Method** | **What You Need** | **How to Transfer to VM** |
| --- | --- | --- | --- |
| ORCA | ORCA Forum (academic) | Forum account, browser | scp from your local machine to VM |
| Gaussian | Licensed distribution | License, vendor access | scp from your local machine to VM |

**Note:**

* Both installers are not publicly downloadable due to licensing restrictions.
* You cannot download these files directly from the VM using wget or curl unless your institution provides a direct, authenticated download link.

after executing the script install\_orca\_gaussian.sh, how can I confirm that the both ORCA and Gaussian are installed properly?

**Copilot said: To confirm that \*\*ORCA\*\* and \*\*Gaussian\*\* are**

To confirm that **ORCA** and **Gaussian** are installed properly after running your install\_orca\_gaussian.sh script, follow these steps:

**1. Confirm ORCA Installation**

**A. Check ORCA executable in your PATH:**

bash

orca --help

**Expected output:**  
You should see the ORCA help message showing available options and usage.

**B. Check ORCA version:**

bash

orca --version

**Expected output:**  
It should display the installed version (e.g., 5.0.4).

**C. (Optional) Run a test calculation:**  
Try running ORCA on a simple input file (e.g., test.inp) to confirm it executes.

**2. Confirm Gaussian Installation**

**A. Check Gaussian environment variables:**

bash

echo $g16root

echo $GAUSS\_SCRDIR

**Expected output:**

* $g16root should show the path to your Gaussian directory.
* $GAUSS\_SCRDIR should show your scratch directory.

**B. Check Gaussian executable in your PATH:**

bash

g16

**Expected output:**  
Running g16 without arguments should show a brief message or an error about missing input, not "command not found".

**C. (Optional) Check Gaussian version:**  
Gaussian doesn't have a typical --version flag, but you can run a very simple job and check the output file for the version string, or run:

bash

g16 < /dev/null

This may print version information.

**3. Troubleshooting**

* If you see “command not found,” make sure you have run source ~/.bashrc in your current shell.
* Double-check that the binaries exist in $HOME/chemsoft/orca-5.0.4 and $g16root/g16.
* For Gaussian, ensure your license and host configuration are set correctly if you see errors about license or host files.

**Quick Summary Table**

| **Software** | **Command** | **Expected Output** |
| --- | --- | --- |
| ORCA | orca --help | ORCA help/usage message |
| ORCA | orca --version | ORCA version (5.0.4) |
| Gaussian | g16 | Not “command not found” (usage or error) |
| Gaussian | echo $g16root | Path to Gaussian install |
| Gaussian | echo $GAUSS\_SCRDIR | Path to Gaussian scratch dir |

**If all commands return expected results, your installations are successful!**