DGX Spark User Guide

NVIDIA Corporation

Oct 14, 2025

Release Notes

This guide is also available for download as a PDF.

Chapter 1. Overview

The DGX Spark is NVIDIA's compact AI computer designed for developers, data scientists, and AI researchers who need powerful computing capabilities for AI development and deployment.

Chapter 2. Getting Started

2.1. DGX Spark Release Notes

This section provides release notes for the DGX Spark, including information about new features, known issues, and software version updates.

2.1.1. Current Software Versions

The following table shows the current version information for the DGX Spark software stack.

Component	Version
Operating System	NVIDIA DGX OS 7.2.3
NVIDIA GPU Driver	580.95.05
NVIDIA CUDA Toolkit	13.0.2

2.1.2. Known Issues

For an updated list of known issues, see *Known Issues*.

2.2. Known Issues

This topic provides a summary of known issues with DGX Spark systems.

2.2.1. Use the supplied power adapter for optimal performance

For optimal performance, use the supplied power adapter with the DGX Spark system. Using a different adapter may reduce performance, prevent boot, or cause unexpected shutdowns.

2.2.2. nvidia-smi reports “Memory-Usage: Not Supported”

On iGPU platforms, `nvidia-smi` will display “Memory-Usage: Not Supported” even though per-process GPU memory is listed. This is expected because iGPUs do not have dedicated framebuffer memory.

2.2.3. Memory reporting differences with unified memory architecture

DGX Spark systems use a unified memory architecture (UMA), where the GPU shares system memory (DRAM) with the CPU and other compute engines. This design reduces latency and allows larger amounts of memory to be used for GPU workloads. On UMA systems, the CPU can dynamically manage DRAM contents, including freeing up memory by swapping pages between DRAM and the system’s SWAP area. However, the *cudaMemGetInfo* API does not account for memory that could potentially be reclaimed from SWAP. As a result, the memory size reported by *cudaMemGetInfo* may be smaller than the actual allocatable memory, since the CPU may be able to release additional DRAM pages by moving them to SWAP.

To more accurately estimate the amount of allocatable device memory on DGX Spark platforms, CUDA application developers should consider the possibility of DRAM reclamation via SWAP and not rely solely on the values returned by *cudaMemGetInfo*. The following provides an example implementation using C standard libraries:

```c
#include <stdio.h>

int getAvailableMemory(long *availableMemoryKb, long *freeSwapKb) {
    FILE *meminfoFile = NULL;
    char lineBuffer[256];
    long hugeTlbTotalPages = -1;
    long hugeTlbFreePages = -1;
    long hugeTlbPageSize = -1;

    if (availableMemoryKb == NULL || freeSwapKb == NULL) {
        return 1;
    }

    meminfoFile = fopen("/proc/meminfo", "r");
    if (meminfoFile == NULL) {
        return 1;
    }

    *availableMemoryKb = -1;
    *freeSwapKb = -1;

    while (fgets(lineBuffer, sizeof(lineBuffer), meminfoFile)) {
        long value;

        if (sscanf(lineBuffer, "MemAvailable: %ld kB", &value) == 1) {
            *availableMemoryKb = value;
        } else if (sscanf(lineBuffer, "SwapFree: %ld kB", &value) == 1) {
            *freeSwapKb = value;
        } else if (sscanf(lineBuffer, "HugePages_Total: %ld", &value) == 1) {
```

(continues on next page)

(continued from previous page)

```
            hugeTlbTotalPages = value;
        } else if (sscanf(lineBuffer, "HugePages_Free: %ld", &value) == 1) {
            hugeTlbFreePages = value;
        } else if (sscanf(lineBuffer, "Hugepagesize: %ld kB", &value) == 1) {
            hugeTlbPageSize = value;
        }

        if (*availableMemoryKb != -1 &&
            *freeSwapKb != -1 &&
            hugeTlbTotalPages != -1 &&
            hugeTlbFreePages != -1 &&
            hugeTlbPageSize != -1) {
            break;
        }
    }

    fclose(meminfoFile);

    if (hugeTlbTotalPages != 0 && hugeTlbTotalPages != -1) {
        *availableMemoryKb = hugeTlbFreePages * hugeTlbPageSize;

        // Hugetlbfs pages are not swappable.
        *freeSwapKb = 0;
    }

    return 0;
}
```

As a workaround for debugging purposes, you can flush the buffer cache manually with the following command:

```
sudo sh -c 'sync; echo 3 > /proc/sys/vm/drop_caches'
```

After flushing the cache, restart your application.

2.3. Hardware Overview

Powered by the NVIDIA Grace Blackwell architecture, DGX Spark enables developers, researchers, and data scientists to prototype, deploy, and fine-tune large AI models on their desktop. This section provides information about the hardware components and specifications.

2.3.1. System Overview

The DGX Spark features:

- NVIDIA Grace Blackwell architecture with integrated GPU and CPU
- 20-core Arm processor with high-performance cores
- 128 GB unified system memory
- Compact desktop form factor (150mm x 150mm x 50.5mm)

- Advanced connectivity including Wi-Fi 7, 10 GbE, and ConnectX-7
- Support for AI models up to 200 billion parameters (or 405B for dual-Spark configuration)

2.3.1.1 Component Descriptions

The DGX Spark includes the following components:

Table 1: Component Specifications

Component	Specification
GPU	NVIDIA Blackwell Architecture with 5th Generation Tensor Cores, 4th Generation RT Cores
CPU	20-core Arm processor (10 Cortex-X925 + 10 Cortex-A725)
Memory	128 GB LPDDR5x unified system memory, 256-bit interface, 4266 MHz, 273 GB/s bandwidth
Storage	1 TB or 4 TB NVMe M.2 with self-encryption
Network	1x RJ-45 (10 GbE), ConnectX-7 Smart NIC, Wi-Fi 7, Bluetooth 5.4
Connectivity	4x USB Type-C, 1x HDMI 2.1a, HDMI multichannel audio
Video Processing	1x NVENC, 1x NVDEC

2.3.2. Physical Specifications

2.3.2.1 Form Factor

- **Chassis Type**: Small form factor (SFF)
- **Dimensions**: 150 mm (L) x 150 mm (W) x 50.5 mm (H)
- **Weight**: 1.2 kg (2.6 lbs)

2.3.2.2 Environmental Requirements

Table 2: Environmental Specifications

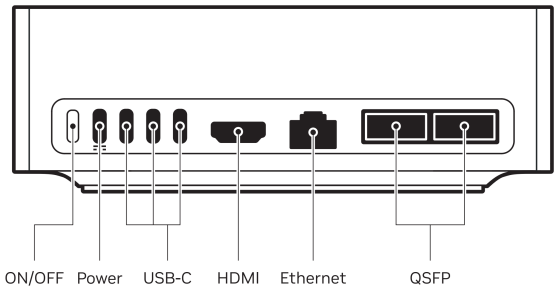
Specification	Value
Ideal Operating Temperature	0°C to 35°C (32°F to 95°F)
Operating Humidity	10% to 90% (non-condensing)
Operating Altitude	Up to 3,000 meters (9,843 feet)

2.3.3. Connectivity and I/O

2.3.3.1 Rear Panel

- Power button

- 4x USB Type-C (one for power delivery)
- 1x HDMI 2.1a display connector
- 1x RJ-45 Ethernet connector (10 GbE)
- 2x QSFP Network connectors (ConnectX-7)



2.3.4. Performance Specifications

2.3.4.1 Compute Performance

- **AI Compute**: Up to 1,000 TOPS (trillion operations per second) inference and up to 1 PFLOP (petaFLOP) at FP4 precision with sparsity
- **CUDA Cores**: 6,144
- **Copy Engines**: 2 (enables simultaneous data transfers to and from GPU memory, improving throughput for AI workloads)
- **CPU Performance**: 20 cores (10 Cortex-X925 + 10 Cortex-A725)
- **Memory Bandwidth**: 273 GB/s
- **Memory Channels**: 16 channels (256 bit) LPDDR5X 8533

2.3.4.2 AI/ML Capabilities

- **Model Support**: AI models up to 200 billion parameters
- **Tensor Performance**: 5th Generation Tensor Cores with FP4 support
- **Framework Support**: TensorFlow, PyTorch, TRT-LLM, and other AI frameworks
- **Use Cases**: Inference, deployment, and fine-tuning of large language models

2.3.5. Power and Thermal Management

2.3.5.1 Power Requirements

- **Power Supply**: 240W external power supply (included)
- **Usage Requirement**: Use of the provided 240W power supply is required for optimal performance. Using a different or lower-rated power supply may result in reduced system performance, failure to boot, or unexpected shutdowns.
- **Input Voltage**: Standard AC power input

2.3.5.2 Thermal Management

- **Cooling Solution**: Integrated thermal management system
- **Form Factor**: Compact design optimized for desktop placement
- **Operating Temperature**: 0°C to 35°C (32°F to 95°F) *Preliminary specifications*

2.4. Initial Setup - First Boot

This guide walks you through setting up your DGX Spark for the first time. You'll choose how to use your system during the setup process and run the first-time setup utility to configure everything. The method you choose will be used throughout the initial setup process. After setup is complete, you can freely switch between desktop and network appliance modes.

2.4.1. What You'll Do

This setup process includes:

- Choosing between desktop or network appliance setup mode
- Preparing your system and connections
- Running the first-time setup utility to configure your system

2.4.2. Choose how to use your system during the setup process

Your DGX Spark can be configured in one of two ways:

Desktop Setup Mode

- Connect keyboard and mouse via USB or Bluetooth
- Work using the Ubuntu desktop directly

Network Appliance Setup Mode

- Access the system remotely over the network
- Use as a server or compute node
- Manage without a local display

> **Note**
>
> The mode you select here will be used throughout the initial setup process. After setup is complete, you can freely switch between desktop and network appliance modes. You're not locked into your original choice.

2.4.3. Get Ready

Important

The DGX Spark device starts up immediately when power is applied. Please attach all peripherals (display, keyboard, mouse, network, etc.) before connecting the power supply.

Before starting, ensure you have:

- Either an Ethernet connection that provides a valid internet connection OR an available Wi-Fi network that provides a valid internet connection without a captive portal (e.g. at a hotel/airport)
- For Desktop Mode: Display, keyboard, and mouse connected (or available via Bluetooth)
- For Network Appliance Mode: A computer on the same network for remote access
- Power connection to the system (the system will start up automatically when power is applied)

Note

Display Troubleshooting: Some displays may have trouble with Spark out of the box. If you are connecting over USB-C/DisplayPort and there is no display, try using HDMI instead.

Note

If you plan to use a wired network connection, plug in the network cable before starting the installation. This helps avoid connection issues later in the process.

2.4.4. Run the First-Time Setup

The first-time setup utility will guide you through:

- Powering on and initializing the system
- Selecting your preferred setup mode
- Downloading and installing critical updates
- Completing your initial configuration

Warning

Critical: Do not shut down or reboot the system during the update process. The installation cannot be interrupted once the download begins, and powering down during updates can cause system damage.

2.4.4.1 Getting Started

The way you start the installation depends on your chosen mode:

Desktop Mode

1. Power on the system
2. The first-time setup utility will start automatically on the connected display
3. Use your wired keyboard and mouse (already connected) to navigate
4. If a keyboard or mouse is not detected, you will be prompted to put your Bluetooth devices in pairing mode:

   USB devices can be plugged in at any time and should start working, even if detected improperly. Bluetooth devices can be put into pairing mode and will generally still pair while on the "Get Started" screen (exception - keyboards that require a passcode to type in won't work on this screen). Once you click on "Get Started," Bluetooth pairing stops, so you will have to power cycle to try again.
5. Follow the on-screen prompts to complete the setup process

Network Appliance Mode

1. Power on the system. This creates a Wi-Fi hotspot that you will use to connect to the system and continue the setup process. The SSID and password for the Wi-Fi hotspot are printed on a sticker attached to the Quick Start Guide included with your DGX Spark's packaging
2. From another computer, connect to the Spark's Wi-Fi hotspot using the SSID and password provided on the Quick Start Guide. A captive portal page will open in the default web browser on your computer. If it does not open automatically, use your browser to navigate to the Spark's system setup page listed on the Quick Start Guide.
3. Follow the on-screen prompts to continue the setup process.

   When the DGX Spark joins your home network, its Wi-Fi hotspot will turn off and your computer will reconnect to the device through your Wi-Fi network to resume the setup process. If you are not able to connect to the DGX Spark after it joins your home network, you must connect a display/keyboard/mouse to continue.

> **Note**
>
> It may take as long as 10 minutes for the DGX Spark to install all of it's updates and join your home network.

2.4.4.2 What to Expect During Setup

The first-time setup utility will guide you through several configuration steps. Simply follow the on-screen prompts to complete each step.

Setup Process Steps:

1. **Language and Time Zone Selection**

   Choose your preferred language and time zone settings for the system. Note that the input fields will filter as you type.
2. **Keyboard Layout Selection** (Desktop Mode only)

   Select your keyboard layout (e.g. US keyboard vs. Russian keyboard). This screen only appears in desktop mode.

3. **Terms and Conditions**

   Review and accept the terms and conditions to continue with the installation.

4. **User Account Creation**

   Create your username and password for system access.

5. **Information Sharing Settings** (Optional)

   Configure analytics and crash reporting preferences. You can skip this step if desired.

6. **Wi-Fi Network Selection**

   Select your Wi-Fi network. This step is automatically skipped if an Ethernet cable that is providing internet access is connected.

7. **Wi-Fi Password**

   Enter the password for your selected Wi-Fi network.

8. **Joining Wi-Fi Network**

   The system connects to your Wi-Fi network and tears down the access point. Your computer will automatically reconnect to your default network.

   Note

   Network Connection Issues:

   - If your computer automatically reconnects to the same network as the Spark, the installation should continue seamlessly
   - If not, you'll need to connect your computer to the same network as the Spark while the setup app is waiting for the network setup process to complete
   - If the setup fails, you must connect a display/keyboard/mouse to continue
   - The modal instructs you to try to reconnect to the Spark's hotspot and try again. This will work if the Spark actually failed to join the network (e.g. wrong password) versus your laptop can't communicate with the Spark
   - If you DO NOT see the hotspot available when this error modal appears, that means the Spark did join the network but your laptop can't communicate with it. This could be because:
     - Device isolation
     - You failed to join the same network as your Spark
     - mDNS does not work on your network due to its configuration (e.g. a complex corporate network)

9. **Software Download and Installation**

   After connecting to the network, the system will automatically download and install the full software image. This process can take some time, and the system may reboot more than once before it's complete. If you are setting up in network appliance mode, you will not be able to access the device during this time. The process may continue for up to 10 minutes after the interface shows that the device is rebooting to finish the setup

> **Warning**
>
> **Do not shut down or reboot the system during this process.** The installation cannot be interrupted once the download begins.

10. **Installation Complete**

    The device will reboot automatically when installation is complete, and you can then use it normally.

2.4.5. Next Steps

Congratulations! Your DGX Spark is now ready to use. Here are the recommended ways to access and start working with your system:

Access Your System

- **Desktop Mode**: Use your DGX Spark like any desktop computer with the monitor, keyboard, and mouse you connected during setup
- **Network Appliance Mode**: Connect to your DGX Spark remotely using the NVIDIA Sync tool. See *NVIDIA Sync* for detailed instructions
- **Dashboard Access**: Use the built-in DGX Dashboard for system monitoring, updates, and JupyterLab access. See *DGX Dashboard* for more information

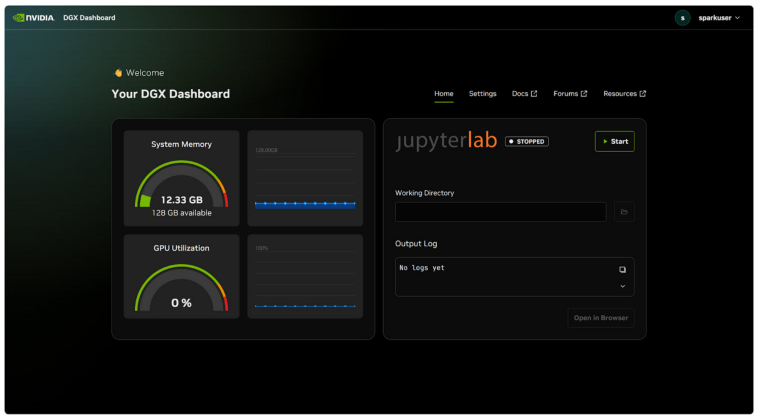


Fig. 1: The DGX Dashboard provides an intuitive interface for system monitoring and development

Additional Resources

- Visit the NVIDIA Spark Developer Portal at https://build.nvidia.com/spark for the latest guides, tutorials, and updates
- Refer to *DGX Spark Release Notes* for the latest software updates and features
- See *Known Issues* for troubleshooting common problems

Your DGX Spark is now ready to power your AI development and deployment workflows!

2.5. System Configuration and Operation

Configuring and operating your DGX Spark effectively is key to delivering consistent results across AI/ML workflows. This section provides an overview of the platform, recommended UEFI settings, clustering procedures, and foundational security guidance to help you deploy, manage, and scale with confidence.

2.5.1. System Overview

Powered by the NVIDIA Grace Blackwell architecture, DGX Spark enables developers, researchers, and data scientists to prototype, deploy, and fine-tune large AI models on their desktop.

2.5.1.1 Flexible Deployment Options

The DGX Spark is designed for maximum flexibility in how you use it:

- **Desktop Mode**: Use with keyboard, mouse, and monitor for local development work
- **Network Appliance Mode**: Operate headless for remote access and server-style deployments

Both modes are fully supported and equally capable, allowing you to choose the setup that best fits your workflow and environment.

2.5.1.2 Key Capabilities

Your DGX Spark enables you to:

- **Run Inference**: Deploy models for real-time AI applications
- **Develop AI Models**: Train and fine-tune models with up to 200 billion parameters
- **Process Data**: Handle large datasets with high-performance computing
- **Experiment Freely**: Test new ideas without cloud computing costs
- **Scale Workloads**: Connect multiple systems for larger projects

2.5.1.3 System Architecture

The DGX Spark is built on NVIDIA's Grace Blackwell architecture, providing:

- **Unified Memory**: 128 GB of high-bandwidth memory for large models
- **High-Performance Computing**: 20-core ARM64-based processor with integrated GPU
- **Advanced Connectivity**: Wi-Fi 7, 10 GbE, CX7 NIC, and multiple I/O options
- **Compact Form Factor**: 150mm x 150mm x 50.5mm desktop design

For detailed hardware specifications, see *Hardware Overview*.

2.5.1.4 Software

Your system comes pre-configured with:

- **NVIDIA DGX OS**: Optimized operating system for AI workloads
- **Development Tools**: CUDA, cuDNN, and NVIDIA's development ecosystem
- **Container Support**: Docker and NVIDIA Container Runtime for easy deployment

- **NGC Integration**: Access to NVIDIA's container registry

For detailed software information, see *Software*.

2.5.1.5 Getting Started

To begin using your DGX Spark:

1. **Initial Setup**: Follow the *Initial Setup - First Boot* to configure your system
2. **Explore Examples**: Try sample workloads to understand capabilities
3. **Configure Development Environment**: Set up your preferred tools and frameworks
4. **Start Building**: Begin your AI development projects

> **Note**
>
> For the most up-to-date tutorials and examples, visit https://build.nvidia.com/spark. This site is regularly updated with new content and serves as the primary resource for practical guides and use cases.

2.5.2. UEFI Settings

This topic provides guidance on accessing and configuring the UEFI settings for the DGX Spark system. While there are no Spark-specific features that require UEFI configuration, you may need to access the UEFI for general system configuration or troubleshooting purposes.

2.5.2.1 Accessing UEFI

To access the UEFI setup menu:

1. Power on or restart the system
2. Immediately press **Esc** or **Del** during the boot process and keep holding it until the UEFI setup menu appears
3. The UEFI setup menu will appear

> **Note**
>
> The timing for pressing Esc or Del is critical. Press the key as soon as the system starts booting, before the operating system begins loading.

2.5.2.2 UEFI Documentation

The DGX Spark system uses an AMI UEFI. For detailed information about UEFI settings and configuration options, refer to the AMI UEFI Manual.

2.5.2.3 General UEFI Configuration

While there are no Spark-specific UEFI requirements, you may need to configure general system settings such as:

- Basic settings (date and time, system language)

- Power-on behavior
- Device boot order
- Security settings (TPM, secure Boot, passwords)
- Network configuration
- RAM disk configuration
-

Important

Always save your UEFI configuration changes before exiting from the **Save and Exit** menu to restart the system with your new settings applied.

2.5.2.4 Troubleshooting

If you experience issues after making UEFI changes:

- Revert to UEFI defaults and restart
- Apply changes incrementally to identify problematic settings
- Update to the latest UEFI version if available
- Consult the AMI UEFI manual for specific setting descriptions
- For additional troubleshooting guidance and support options, see spark-maintenance-troubleshooting

2.5.3. Spark Clustering

2.5.3.1 Connecting DGX Spark Systems into a Virtual Cluster

2.5.3.1.1 Overview

This guide explains how to connect two DGX Spark systems into a virtual compute cluster using simplified networking configuration and a QSFP/CX7 cable for high-performance interconnect.

The goal is to enable distributed workloads across Grace Blackwell GPUs using MPI (for inter-process CPU communication) and NCCL v2.28.3 (for GPU-accelerated collective operations).

2.5.3.1.2 System Requirements

Before you begin, ensure the following:

- Both DGX Spark systems have Grace Blackwell GPUs, are connected to each other using a QSFP/CX7 cable, and are running Ubuntu 24.04 (or later) with NVIDIA drivers installed
- The systems have internet access for initial software setup
- You have sudo/root access on both systems

2.5.3.1.3 Setup Networking Between Spark Systems

2.5.3.1.3.1 Option 1: Automatic IP assignment (recommended)

Follow these steps on **both DGX Spark nodes** to configure network interfaces using `netplan`:

1. **Create the netplan configuration file**

```
sudo tee /etc/netplan/40-cx7.yaml > /dev/null <<EOF
network:
  version: 2
  ethernets:
    enp1s0f0np0:
      link-local: [ ipv4 ]
    enp1s0f1np1:
      link-local: [ ipv4 ]
EOF
```

2. **Set appropriate permissions on the configuration file**

```
sudo chmod 600 /etc/netplan/40-cx7.yaml
```

3. **Apply the netplan configuration**

```
sudo netplan apply
```

2.5.3.1.3.2 Option 2: Manual IP assignment (advanced)

Follow these steps to manually assign IP addresses for dedicated cluster networking.

On Node 1:

1. **Assign a static IP address and bring up the interface**

```
sudo ip addr add 192.168.100.10/24 dev enP2p1s0f1np1
sudo ip link set enP2p1s0f1np1 up
```

On Node 2:

1. **Assign a static IP address and bring up the interface**

```
sudo ip addr add 192.168.100.11/24 dev enP2p1s0f1np1
sudo ip link set enP2p1s0f1np1 up
```

Verify connectivity:

- From **Node 1**, test connection to Node 2:

```
ping -c 3 192.168.100.11
```

- From **Node 2**, test connection to Node 1:

```
ping -c 3 192.168.100.10
```

2.5.3.1.4 Run the DGX Spark Discovery Script

This step will automatically identify interconnected DGX Spark systems and set up SSH authentication without requiring a password.

On both nodes:

1. **Run the discovery script**

```
./discover-sparks
```

Example output:

```
Found: 192.168.100.10 (spark-1b3b.local)
Found: 192.168.100.11 (spark-1d84.local)

Copying your SSH public key to all discovered nodes using ssh-copy-id.
You may be prompted for your password on each node.
Copying SSH key to 192.168.100.10 ...
Copying SSH key to 192.168.100.11 ...
nvidia@192.168.100.11's password:

SSH key copy process complete. These two sparks can now talk to each
↪other.
```

2.5.3.1.5 Install Required Software

To support distributed workloads, both systems must have MPI (for CPU process communication) and NCCL (for GPU collective communication) installed.

1. **Install MPI (OpenMPI)**

MPI allows distributed processes across systems to communicate.

```
sudo apt update
sudo apt install -y openmpi-bin libopenmpi-dev
```

2. **Install NCCL v2.28.3 (or later)**

NCCL provides fast GPU-to-GPU communication over QSFP/CX7 and supports multi-rail socket interfaces. The instructions below are for NCCL v2.28.3. Visit https://developer.nvidia.com/nccl/nccl-download for the latest version.

```
wget https://developer.download.nvidia.com/compute/machine-learning/repos/
↪ubuntu2004/x86_64/libnccl2_2.8.3-1+cuda11.2_amd64.deb
wget https://developer.download.nvidia.com/compute/machine-learning/repos/
↪ubuntu2004/x86_64/libnccl-dev_2.8.3-1+cuda11.2_amd64.deb

sudo dpkg -i libnccl2_2.8.3*.deb libnccl-dev_2.8.3*.deb
```

3. **Build the NCCL test suite**

These tools verify GPU-to-GPU communication and measure performance across nodes.

```
git clone https://github.com/NVIDIA/nccl-tests
cd nccl-tests
make MPI=1
```

2.5.3.1.6 Run a Test Workload

Now that networking and software are configured, run an all-reduce benchmark to test NCCL and MPI across nodes.

1. **Run the test from either machine**

   This example sets the LD_LIBRARY_PATH environment to include the locations of the OpenMPI and CUDA libraries. Adjust this path as necessary to match your system configuration.

   ```
   export LD_LIBRARY_PATH='/usr/local/openmpi/lib:/usr/local/cuda/lib64'
   mpirun -np 2 -H 192.168.100.11:1,192.168.100.21:1 \
     -bind-to none -map-by slot \
     -x NCCL_DEBUG=INFO -x LD_LIBRARY_PATH \
     -mca oob_tcp_if_exclude docker0,lo,virbr0,bmc_redfish0,wlP9s9 \
     -mca btl_tcp_if_exclude docker0,lo,virbr0,bmc_redfish0,wlP9s9 \
     ./build/all_reduce_perf -b 8 -e 512M -f 2 -g 1
   ```

   This uses one GPU on each system to perform a collective reduce operation and report bandwidth and latency.

2.5.3.1.7 Troubleshooting

- Ensure the QSFP/CX7 interface is active and used for IP assignment
- Verify connectivity between nodes via `ping`
- Use `nvidia-smi` to confirm GPU status
- Use `NCCL_DEBUG=INFO` to show detailed diagnostics during the test
- Check your interface bindings with `ip a` and `ethtool`
- If the discovery script fails, manually verify SSH connectivity between nodes
- For additional troubleshooting guidance and support options, see spark-maintenance-troubleshooting

2.5.3.1.8 Next Steps

Once tested, this configuration can be scaled to support:

- Job orchestration with Slurm or Kubernetes
- Containerized execution with Singularity or Docker

2.6. Software

The DGX Spark comes with a comprehensive software stack optimized for AI development, machine learning, and data science workflows. This section provides detailed information about the included software components and their configuration.

2.6.1. DGX OS

2.6.1.1 Overview

NVIDIA DGX OS is a customized Linux distribution that provides a stable, tested, and supported operating system foundation for running AI, machine learning, and analytics applications on DGX systems. It includes platform-specific optimizations, drivers, and diagnostic tools tailored for NVIDIA hardware.

DGX OS serves as the underlying operating system for your DGX Spark, providing:

- A robust Linux foundation optimized for AI workloads
- Pre-configured drivers and system settings for NVIDIA hardware
- Security updates and system maintenance capabilities
- Compatibility with the broader NVIDIA software ecosystem

Note

For more information about DGX OS, see the official documentation at: https://docs.nvidia.com/dgx/dgx-os-7-user-guide/introduction.html

2.6.1.2 Release Cadence

DGX OS follows a regular release schedule with updates typically provided twice per year, around February and August, for the first two years after initial release. Additional updates and security patches are provided between major releases and throughout the support lifecycle.

2.6.2. NVIDIA Sync

2.6.2.1 Overview

NVIDIA Sync is a system tray utility that provides a simple way of accessing your DGX Spark system from another machine when it's running as a headless appliance (without a monitor or keyboard).

2.6.2.2 Installation

1. Download the latest version of NVIDIA Sync from https://build.nvidia.com/spark/connect-to-your-spark/sync. Installers are available for Windows, macOS, and Linux.
2. Run the installer.
3. NVIDIA Sync will search for compatible applications that can connect remotely to the DGX Spark. Select the applications that you want to use, and click **Next**.
4. Provide the DGX Spark's machine name and your login credentials.

The selected applications will be automatically configured and listed in the NVIDIA Sync system tray icon.

2.6.2.2.1 Supported Applications

- AI Workbench
- Cursor IDE

- VSCode

2.6.2.2.2 Additional Connection Methods

- DGX Dashboard (via web browser)
- SSH Terminal (SSH keys automatically managed by NVIDIA Sync)

2.6.3. DGX Dashboard

2.6.3.1 Overview

The DGX Spark comes with a built-in dashboard that provides an overview of the system's current operational metrics, the ability to apply updates, change some system settings, and access local Jupyter Notebooks.

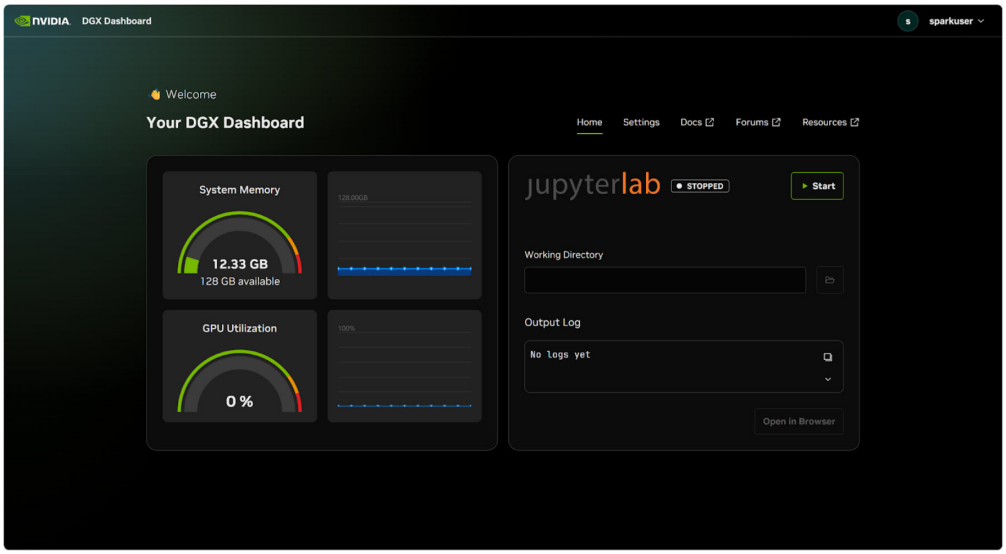


Fig. 2: The DGX Dashboard provides real-time system monitoring and integrated JupyterLab access

> **Note**
>
> To run updates and change the device name, you must have *sudo* access. The account created during initial setup will have access.

2.6.3.2 Integrated JupyterLab

The dashboard includes an integrated JupyterLab instance that provides a convenient development environment:

- When started, JupyterLab creates a virtual environment in the specified working directory and automatically installs a set of recommended packages
- If you enter a new working directory and start JupyterLab, a new environment will be created
- Each user account on the device is assigned a port located in */opt/nvidia/dgx-dashboard-service/jupyterlab_ports.yaml*

- To access JupyterLab remotely, you must tunnel it just like the dashboard itself. The port to tunnel is in the ports file. Using NVIDIA Sync, this tunnel is managed for you automatically and just works

2.6.3.3 Accessing the Dashboard

The dashboard can be accessed locally by clicking on the “Show Apps” button in the bottom left corner of the Ubuntu desktop. Then, in the app grid, select the “DGX Dashboard” shortcut to open the dashboard in your default web browser.

Remotely, the dashboard can be accessed using NVIDIA Sync or via a manually created SSH tunnel. If using NVIDIA Sync, after connecting, simply click on the “DGX Dashboard” button and the dashboard will open in your default web browser at *http://localhost:11000*.

To manually access over SSH, first open a tunnel, e.g., *ssh -L 11000:localhost:11000 <username>@<IP or spark-abcd.local>*. Then, open the dashboard in your web browser at *http://<spark-host-ip>:11000*.

2.6.4. NVIDIA Container Runtime for Docker

2.6.4.1 Overview

The NVIDIA Container Runtime enables Docker containers to access GPU resources on DGX Spark systems. This runtime acts as a bridge between Docker and the NVIDIA drivers, allowing containers to utilize GPU acceleration for AI/ML workloads, CUDA applications, and other GPU-accelerated software.

Key benefits: - Seamless GPU access within containers - Automatic driver and library management - Support for multi-GPU configurations - Compatibility with popular container orchestration platforms

The runtime works in conjunction with the NVIDIA Container Toolkit, which provides the necessary components to expose GPU devices and CUDA libraries to containerized applications.

2.6.4.2 Installation

The NVIDIA Container Toolkit is preinstalled and configured on DGX Spark systems. This includes:

- NVIDIA Container Runtime
- Docker integration
- GPU device access configuration
- CUDA library management

The runtime is ready to use out of the box for running GPU-accelerated containers.

2.6.4.2.1 Optional: Add User to Docker Group

By default, Docker requires *sudo* privileges to run commands. Adding your user to the docker group allows you to run Docker commands without *sudo*, which provides:

- **Convenience**: No need to type *sudo* before every Docker command
- **Better workflow**: Seamless integration with development tools and scripts
- **Reduced friction**: Faster iteration when working with containers

To add your user to the docker group:

```
sudo usermod -aG docker $USER
```

Important: You must log out and log back in (or restart your session) for the group membership to take effect.

Note: This step is optional. You can continue using Docker with *sudo* if you prefer not to modify group memberships.

2.6.4.3 Usage

2.6.4.3.1 Basic GPU Access

Run a container with GPU access using the *–gpus* flag:

```
docker run -it --gpus=all nvcr.io/nvidia/cuda:13.0.1-devel-ubuntu24.04 nvidia-
↪smi
```

This command: - Runs an interactive container (*-it*) - Enables access to all GPUs (*–gpus=all*) - Uses the NVIDIA CUDA development image - Executes *nvidia-smi* to display GPU information

2.6.4.3.2 Set GPU Capabilities

Control which GPU capabilities are available to the container:

```
docker run -it --gpus '"capabilities=compute,utility"' nvcr.io/nvidia/cuda:13.
↪0.1-devel-ubuntu24.04 nvidia-smi
```

2.6.4.3.3 Mount CUDA Libraries

For applications that need specific CUDA libraries, mount them from the host:

```
docker run -it --gpus=all \
  -v /usr/local/cuda:/usr/local/cuda:ro \
  nvcr.io/nvidia/cuda:13.0.1-devel-ubuntu24.04 bash
```

2.6.4.4 Validation

2.6.4.4.1 Test GPU Access

1. Run the test command to verify GPU access:

   ```
   docker run -it --gpus=all nvcr.io/nvidia/cuda:13.0.1-devel-ubuntu24.04
   ↪nvidia-smi
   ```

   Expected output should show: - GPU device information - Driver version - CUDA version - Memory usage and temperature

2. Check runtime configuration:

   ```
   docker info | grep -A 10 "Runtimes"
   ```

3. Verify NVIDIA runtime is available:

   ```
   docker run --rm --runtime=nvidia nvcr.io/nvidia/cuda:13.0.1-devel-
   ↪ubuntu24.04 nvidia-smi
   ```

2.6.4.4.2 Inspect Container GPU Access

Check what GPU resources are available inside a running container:

```
docker run -it --gpus=all nvcr.io/nvidia/cuda:13.0.1-devel-ubuntu24.04 bash
# Inside the container:
nvidia-smi
ls /dev/nvidia*
```

2.6.4.5 Troubleshooting

2.6.4.5.1 Runtime Not Found

If you encounter "runtime not found" errors:

1. Verify NVIDIA Container Toolkit is installed:

   ```
   nvidia-ctk --version
   ```

2. Check Docker daemon configuration:

   ```
   cat /etc/docker/daemon.json
   ```

3. Restart Docker service:

   ```
   sudo systemctl restart docker
   ```

2.6.4.5.2 Driver/Container CUDA Mismatch

If you see CUDA version mismatches:

1. Check host CUDA driver version:

   ```
   nvidia-smi
   ```

2. Use a container image with compatible CUDA version:

   ```
   docker run -it --gpus=all nvcr.io/nvidia/cuda:12.0.1-devel-ubuntu24.04
   ↪nvidia-smi
   ```

2.6.4.5.3 Permission Issues

If you encounter permission errors:

1. Ensure your user is in the docker group (if not using sudo):

   ```
   groups $USER
   ```

2. Check device permissions:

   ```
   ls -la /dev/nvidia*
   ```

3. Verify Docker daemon has access to GPU devices:

   ```
   sudo docker run -it --gpus=all nvcr.io/nvidia/cuda:13.0.1-devel-ubuntu24.
   ↪04 nvidia-smi
   ```

2.6.4.5.4 Container Startup Issues

If containers fail to start:

1. Check Docker logs:

```
docker logs <container_id>
```

2. Verify GPU devices are available on host:

```
ls /dev/nvidia*
```

3. Test with a minimal container:

```
docker run --rm --gpus=all nvcr.io/nvidia/cuda:13.0.1-devel-ubuntu24.04
↪echo "GPU test successful"
```

2.6.5. NGC

2.6.5.1 Overview

NVIDIA GPU Cloud (NGC) is a comprehensive registry of GPU-optimized containers, pre-trained models, and AI/ML software that enables rapid development and deployment of AI applications. For DGX Spark users, NGC provides access to the latest frameworks, tools, and optimized environments specifically designed for the Grace Blackwell architecture.

Key benefits for DGX Spark users:

- **Optimized Containers**: Pre-configured environments with the latest AI/ML frameworks, CUDA, and libraries optimized for Grace Blackwell GPUs
- **Pre-trained Models**: Access to state-of-the-art models and model collections for various AI tasks
- **Rapid Development**: Skip complex environment setup and focus on your AI/ML projects
- **Cutting-edge Software**: Access to the latest NVIDIA software stack and experimental features

NGC is particularly valuable for DGX Spark users because it provides the most current and optimized software stack for this new platform, ensuring you have access to the latest performance optimizations and features.

2.6.5.2 Getting Started

2.6.5.2.1 Create an NGC Account

1. Visit the NGC website
2. Click **Sign Up** and create a free account
3. Verify your email address
4. Complete your profile information

2.6.5.2.2 Generate an API Key

1. Log in to your NGC account
2. Navigate to **Setup** -> **API Key**
3. Click **Generate API Key**

4. Copy and securely store your API key

> **Note**
>
> Your API key is required for pulling containers and accessing NGC resources. Keep it secure and never share it publicly.

2.6.5.2.3 Install NGC CLI (Optional)

The NGC CLI provides convenient command-line access to NGC resources:

```
# Download and install NGC CLI
wget https://ngc.nvidia.com/downloads/ngccli_linux.zip
unzip ngccli_linux.zip
echo "export PATH=\"\$PATH:$(pwd)/ngc-cli\"" >> ~/.bash_profile && source ~/.
↪bash_profile
ngc config set
```

2.6.5.2.4 Authenticate with Docker

Configure Docker to access NGC registries:

```
# Login to NGC with Docker
docker login nvcr.io
# Username: $oauthtoken
# Password: <your-api-key>
```

2.6.5.3 Basic Usage

2.6.5.3.1 Pull and Run a Container

Start with a popular AI/ML framework container:

```
# Pull a PyTorch container optimized for Grace Blackwell
docker pull nvcr.io/nvidia/pytorch:24.08-py3

# Run the container with GPU access
docker run -it --gpus=all nvcr.io/nvidia/pytorch:24.08-py3
```

2.6.5.3.2 Explore Available Resources

Browse NGC resources through the web interface:

- **Containers**: AI/ML frameworks, development environments, and specialized tools
- **Models**: Pre-trained models for computer vision, natural language processing, and more
- **Helm Charts**: Kubernetes deployment configurations
- **Jupyter Notebooks**: Interactive tutorials and examples

2.6.5.4 Common Workflows

2.6.5.4.1 Development Environment

Use NGC containers as your development environment:

```
# Run a development container with persistent storage
docker run -it --gpus=all \
  -v /path/to/your/project:/workspace \
  nvcr.io/nvidia/pytorch:24.08-py3
```

2.6.5.4.2 Model Inference and Training

Access pre-trained models and training scripts:

```
# Pull a model from NGC
ngc registry model download-version nvidia/bert-base-uncased:1

# Or use models directly in containers
docker run -it --gpus=all \
  nvcr.io/nvidia/tensorflow:24.08-tf2-py3
```

2.6.5.5 Best Practices

2.6.5.5.1 Container Management

- **Pin Versions**: Use specific container tags for reproducible environments
- **Regular Updates**: Periodically update to newer container versions for latest optimizations
- **Resource Limits**: Set appropriate memory and CPU limits for your workloads

2.6.5.5.2 Data Persistence

- **Volume Mounts**: Mount your data directories into containers for persistence
- **Model Storage**: Store trained models and checkpoints outside containers
- **Configuration**: Keep configuration files in version control

2.6.5.5.3 Security

- **API Key Security**: Store your NGC API key securely and rotate it regularly
- **Container Scanning**: Scan containers for vulnerabilities before use
- **Network Security**: Use appropriate network configurations for your environment

2.6.5.6 Troubleshooting

2.6.5.6.1 Common Issues

Authentication Failures

```
# Verify your API key is correct
docker login nvcr.io
# Check if your account has access to the requested resource
```

Container Pull Issues

```
# Check network connectivity
ping nvcr.io

# Verify container name and tag
docker search nvcr.io/nvidia/
```

GPU Access Problems

```
# Verify NVIDIA Container Runtime is installed
docker run --rm --gpus=all nvidia/cuda:12.0-base-ubuntu20.04 nvidia-smi
```

2.6.5.6.2 Getting Help

- **NGC Documentation**: Visit the NGC documentation
- **Community Forums**: Join the NVIDIA Developer Forums
- **Additional Support**: For troubleshooting guidance and support options, see spark-maintenance-troubleshooting

2.7. Common Use Cases

The DGX Spark is designed to support a wide range of AI, machine learning, and data science workflows. Visit https://build.nvidia.com/spark for practical guides to help you get started. That site is regularly updated with new content and information and will be the single source of truth for your Spark device.

2.8. OS and Component Update Guide

This section provides guidance for updating the operating system, software components, and firmware on your DGX Spark. The DGX Spark runs on NVIDIA DGX OS, which is an Ubuntu-based Linux distribution optimized for AI workloads.

> **Note**
>
> **Founders Edition Only**: The update information in this guide applies only to the DGX Spark Founders Edition. Devices from other manufacturers may have different update procedures.

2.8.1. Update Methods

We strongly recommend using the DGX Dashboard for all system updates on your |spark|. The dashboard ensures your system stays up to date with the latest NVIDIA-optimized components and configurations. While standard Ubuntu package management methods will work, the dashboard provides the most reliable and tested update path for your DGX system.

2.8.2. Using DGX Dashboard for Updates

The DGX Dashboard is the **primary and recommended** way to perform system updates on your DGX Spark. It provides a centralized interface for:

- Viewing available system updates
- Installing security patches and system updates
- Managing NVIDIA driver updates
- Managing firmware updates
- Monitoring update status and progress

For detailed information about using the DGX Dashboard, including how to access it and perform updates, see *DGX Dashboard*.

Warning

Before performing any system or firmware updates:

- Ensure your system is connected to a stable power source
- Close all running applications and save your work
- Have a recovery plan in place
- Schedule updates during maintenance windows when possible

2.8.3. Manual System Updates

Note: While manual updates using standard Ubuntu package management commands will work, we strongly recommend using the DGX Dashboard for all updates to ensure optimal system performance and compatibility.

For advanced users or when the DGX Dashboard is not available, you can perform system updates manually using the following steps:

1. Open a remote or local terminal on the DGX Spark device.
2. Run the following commands:

```
sudo apt update
sudo apt dist-upgrade
sudo fwupdmgr refresh
sudo fwupdmgr upgrade
sudo reboot
```

These commands will:

- Update the package lists and upgrade all installed packages (including OS components and drivers)
- Refresh the firmware metadata and upgrade all firmware components
- Reboot the system to apply updates

2.8.4. Update Best Practices

- **Use the DGX Dashboard**: Always prefer the DGX Dashboard for system updates to ensure compatibility and optimal performance
- **Regular updates**: Check for updates regularly, especially security patches
- **Backup before major updates**: Always backup critical data before major system changes
- **Stable power**: Ensure your system has a stable power supply during updates
- **Maintenance windows**: Schedule updates during planned maintenance windows when possible

2.9. System Recovery

This section provides information about system recovery procedures for your DGX Spark.

> **Note**
>
> **Founders Edition Only**: This recovery information applies only to the DGX Spark Founders Edition. Devices from other manufacturers may have different recovery procedures.

2.9.1. System Recovery Overview

The system recovery process for the DGX Spark allows you to restore the operating system and firmware to their original factory state. This process is useful when dealing with system corruption, fatal configuration errors, or other issues that prevent normal system operation.

2.9.1.1 Recovery Requirements

Before beginning the recovery process, ensure you have the following:

- A USB flash drive with 16GB or larger capacity
- A keyboard and display connected to your DGX Spark
- Access to download the recovery media from NVIDIA support

2.9.1.2 Recovery Process Steps

The recovery process involves downloading recovery media, creating a bootable USB drive, and following a series of UEFI configuration steps. Follow these steps carefully to ensure successful system recovery.

1. **Download Recovery Media**:
   - Download the recovery media archive file (tar.gz format) from the NVIDIA support website
   - Save the file to your local system
2. **Create Recovery USB Drive**:
   - Download and unzip the recovery media archive file
   - Insert the USB drive into your system
   - Within the extracted files, use one of the following commands to create the recovery drive:

Windows:

```
CreateUSBKey.cmd
```

Linux:

```
CreateUSBKey.sh
```

MacOS:

```
CreateUSBKeyMacOS.sh
```

> **Warning**
>
> This process will erase all data on your USB drive. Ensure you have backed up any important data before proceeding.

3. **Boot from Recovery USB Drive**:
   - Connect the USB drive to one of the USB ports on your DGX Spark
   - If your DGX Spark is powered off, boot the device into UEFI settings by holding down the `Del` key immediately after powering on the device
4. **Restore UEFI Defaults**:
   - Tap the `Right Arrow` key to select the “Save & Exit” page within the UEFI settings
   - Tap `Down Arrow` to select “Restore Defaults” and select “Yes” in response to “Load Optimized Defaults”
   - Select “Save Changes and Reset” - the device will reboot
   - While the device is rebooting, hold down the `Del` key to re-enter UEFI settings a second time
5. **Enable Secure Boot**:
   - Use the `Right Arrow` key to select “Security”
   - Confirm that “Secure Boot” is set to “Enabled”
   - Select “Restore Factory Keys”
   - Use the arrow keys to select the “Save and Exit” BIOS screen, and select “Save Changes and Reset”
6. **Boot from Recovery Media**:
   - While the device is rebooting, hold down the `Del` key to re-enter UEFI settings a third time
   - Tap the `Right Arrow` key to select the “Save & Exit” page in UEFI settings
   - Tap `Down Arrow` to move to the “Boot Override” section
   - Select the USB drive and tap `Enter`
   - The device will reboot using the USB drive - follow the on-screen steps to update your firmware and re-install the OS on the device’s SSD
7. **Restore System from Recovery Environment**

- On the welcome screen, press `Enter` to continue. To cancel the process, press Esc.

- On the warning screen, select [EXIT] to restart without proceeding, or select [START RECOVERY] to begin reflashing the SSD. Be aware that this will completely erase the internal SSD on the DGX Spark.

- Monitor the Recovery Progress screen for detailed output from the recovery process and the progress of the SSD reflash.

- When the process is complete, review the progress screen and follow the prompt to continue. Scroll or page through the recovery output if needed, for example during customer support.

- On the final screen, confirm that the DGX Spark has been reset to factory state and press `Enter` to restart the device.

Copyright