# Contents

[Contents 1](#_Toc295813250)

[About GPU add-on 1](#_Toc295813251)

[Scope 1](#_Toc295813252)

[Configuration to enable <GPU add-on> 2](#_Toc295813253)

[1. Prerequisites 2](#_Toc295813254)

[2. Installation files 2](#_Toc295813255)

[3. Installation 3](#_Toc295813256)

[3.1 Install package on management host 3](#_Toc295813257)

[3.2 Install package on all compute hosts at once 4](#_Toc295813258)

[3.3 Restart the cluster 5](#_Toc295813259)

[4. Configuration 5](#_Toc295813260)

[5. Verification 5](#_Toc295813261)

[6. New metrics description 6](#_Toc295813262)

[7. Uninstall Instructions 7](#_Toc295813263)

# About GPU add-on

The goal of the GPU add-on is to enable the utilization of GPU devices within Symphony seamlessly. It provides the ability to monitor and schedule GPU devices as native Symphony resources such as CPUs.

# Scope

|  |  |
| --- | --- |
| Applicability | |
| Operating system | Linux 2.6 glib 2.3 x64 bits |
| Symphony version | Symphony 5.1 |
| Dependencies | |
| File system | N/A |
| <Other> | This feature will work when the following software is installed:   * Management host:   symgpu-linux2.6-glibc2.3-x86\_64-5.1.0-\*.rpm   * Compute host:   symgpupackage-linux2.6-glibc2.3-x86\_64-5.1.0-\*.tar.gz   * To develop applications, the DE installation should be patched with symgpusample-linux2.6-glibc2.3-x86\_64-5.1.0-\*.tar.gz |
| Limitations | |
| <Limitation> | 1. Standby services have limited usage with GPU integration feature. A service instance in standby mode will still occupy the GPU devices. For more information about standby service feature, refer to Symphony documentation.  2. Current scheduling assigns tasks to GPU devices exclusively even if shared mode devices are allowed (SI\_GPU\_COMPUTE\_MODE=0 or omitting the variable).  3. maxOtherInstances feature is not supported with GPU integration. For more information about maxOtherInstances feature, refer to Symphony documentation.  4. Messages related to onGpuInvoke (...) may be reported as onInvoke (...) as the integration is done in the application service. |
| Known Issues | N/A |

# Configuration to enable <GPU add-on>

## Prerequisites

* + NVIDIA CUDA Developer Drivers for Linux 64 bits ver.3.2 or later on all GPU-enabled compute hosts. Assumptions:
    1. CUDA installation directories:
       1. /usr/lib[64] for Cuda and OpenCL libraries
       2. /usr/local/cuda/lib[64] for CUDA SDK installation
    2. LIBC installation directories:
       1. /lib[64] for libc library
  + Symphony Version 5.1 on Linux 2.6 glib 2.3 x64 bits.
  + Symphony DE 5.1 on your build machine to create a build environment.

## Installation files

| File name | Description |
| --- | --- |
| symgpu-linux2.6-glibc2.3-x86\_64-5.1.0-\*.rpm | File list (names in bold indicate what will be added):   * $EGO\_CONFDIR/ego.conf * $EGO\_CONFDIR/ego.shared * $EGO\_CONFDIR/ego.cluster.<clustername> * $EGO\_SERVERDIR/vemkd * $EGO\_SERVERDIR/lim * **$**EGO\_SERVERDIR/**elim.nvidia** * $EGO\_LIBDIR/**libpccgpu.so** * $SOAM\_HOME/5.1/linux2.6-glibc2.3-x86\_64/lib/ **libpccgpu.so** * $SOAM\_HOME/5.1/linux2.6-glibc2.3-x86\_64/lib/ **libsoamgpuservicecontainer.so** * $SOAM\_HOME/5.1/linux2.6-glibc2.3-x86\_64/lib64/ **libpccgpu.so** * $SOAM\_HOME/5.1/linux2.6-glibc2.3-x86\_64/lib64/ **libsoamgpuservicecontainer.so** * $SOAM\_BINDIR/**soamgpureader** * $SOAM\_BINDIR/**GpuTestService** (installed if host has GPU devices) * $SOAM\_BINDIR/**GpuTestClient** * $SOAM\_HOME/conf/**GpuTestApp.xml** * $SOAM\_HOME/docs/**GPUREADME** |
| symgpupackage-linux2.6-glibc2.3-x86\_64-5.1.0-\*.tar.gz | File list:   * $EGO\_SERVERDIR/ lim * **$**EGO\_SERVERDIR/**elim.nvidia** * $EGO\_LIBDIR/**libpccgpu.so** * $SOAM\_HOME/5.1/linux2.6-glibc2.3-x86\_64/lib/ **libpccgpu.so** * $SOAM\_HOME/5.1/linux2.6-glibc2.3-x86\_64/lib/ **libsoamgpuservicecontainer.so** * $SOAM\_HOME/5.1/linux2.6-glibc2.3-x86\_64/lib64/ **libpccgpu.so** * $SOAM\_HOME/5.1/linux2.6-glibc2.3-x86\_64/lib64/ **libsoamgpuservicecontainer.so** * $SOAM\_BINDIR/**soamgpureader** * $SOAM\_BINDIR/**GpuTestService** (installed if host has GPU devices) * $SOAM\_BINDIR/**GpuTestClient** * $SOAM\_HOME/conf/**GpuTestApp.xml** |
| symgpusample-linux2.6-glibc2.3-x86\_64-5.1.0-\*.tar.gz | File list:   * $SOAM\_HOME/5.1/linux2.6-glibc2.3-x86\_64/lib/ **libpccgpu.so** * $SOAM\_HOME/5.1/linux2.6-glibc2.3-x86\_64/lib/ **libsoamgpuservicecontainer.so** * $SOAM\_HOME/5.1/linux2.6-glibc2.3-x86\_64/lib64/ **libpccgpu.so** * $SOAM\_HOME/5.1/linux2.6-glibc2.3-x86\_64/lib64/ **libsoamgpuservicecontainer.so** * $SOAM\_BINDIR/**soamgpureader** * $SOAM\_HOME/5.1/samples/CPP/GpuSampleApp – directory containing 64-bit application * $SOAM\_HOME/5.1/samples/CPP/GpuSampleApp – directory containing 32-bit application |

## Installation

### 3.1 Install package on management host

1. Download the installation file symgpu-linux2.6-glibc2.3-x86\_64-5.1.0-\*.rpm.
2. Source the Symphony environment. Default install location is /opt/ego:

$ cd /opt/ego # cd to install location

$ source cshrc.platform # csh

-or-

$ . ./profile.platform # sh

1. Optionally you can set environment variable "GPU\_MAX\_DEVICE\_NUM <num>", where <num> is the number of GPU devices on the compute host; if there are different numbers on different hosts, insert the largest number (8 is the maximum number of GPU devices currently supported by Symphony). It will configure the number of GPU devices, and their respective metrics will be displayed in the system. If this variable is not set, the default number will be 8.

$ setenv GPU\_MAX\_DEVICE\_NUM <num> # csh

-or-

$ export GPU\_MAX\_DEVICE\_NUM=<num> # sh

1. Install the rpm package in the directory where the Symphony cluster was installed, using the same user to install the rpm; for example, if Symphony was installed using sudo:

$ sudo rpm -ivh symgpu-linux2.6-glibc2.3-x86\_64-5.0.0-\*.rpm \

--prefix $EGO\_TOP –force

1. Optionally, you can restart the cluster as described in section 3.3; however you can postpone the restart until the GPU add-on is installed on the compute hosts.

### 3.2 Install package on all compute hosts at once

1. Download the installation file symgpupackage-linux2.6-glibc2.3-x86\_64-5.1.0-\*.tar.gz.
2. Source the Symphony environment. Default install location is /opt/ego:

$ cd /opt/ego # cd to install location

$ source cshrc.platform # csh

-or-

$ . ./profile.platform # sh

1. Log on to Symphony as Cluster Administrator:

$ egosh user logon

By default, the username/password are Admin/Admin, respectively.

1. Add the deployment package to Symphony's repository:

$ rsdeploy add gpupack –p symgpupackage-linux2.6-glibc2.3-x86\_64-5.1.0-\*.tar.gz

1. Install the package on the hosts belonging to the target GPU resource group. ComputeHosts is the default resource group. HINT: use "egosh rg" to get a list of resource groups.

$ rsdeploy install gpupack -r ComputeHosts

Omit the -r option to install on all hosts - not recommended as management hosts will also be selected.

Get the install status to check for possible errors:

$ rsdeploy status gpupack

### 3.3 Restart the cluster

1. Log on to Symphony as Cluster Administrator:

$ egosh user logon

1. Shut down the WEBGUI service:

$ egosh service stop WEBGUI

1. Restart the cluster:

$ egosh ego restart all

## Configuration

N/A

## Verification

Verification is performed by configuring and running the Test Application.

1. Configure a Symphony GPU Resource Group and add GPU hosts to this group. For more details on new Resource Group creation, see the Resource Sharing and Distribution section in the Symphony Cluster and Application Management Guide. In case a GPU-enabled host will be used at the same time for both GPU- and CPU-related workload, that host could also remain in the default ComputeHosts group; otherwise, the host should be removed from the default ComputeHosts group. If all compute hosts in the cluster have GPU devices and no devices are set to SHARED compute mode, then skip the new Resource Group creation and modify the slot calculation of ComputeHosts Resource Group.
2. Set the number of slots per host in the Group equal to the number of exclusive GPU devices attached:
   1. From the "Configure Resource Groups" tab, select the resource group link you want to modify (either the one created or "ComputeHosts").
   2. Select the "Advanced Formula" radio button and enter "gpuexclusive\_thread".
3. Start the Symphony PMC to create a consumer:
   1. Log on to the PMC (by default Admin/Admin).
   2. From quick links, select Consumers.
   3. In global action, choose 'Create a Consumer' and specify "GpuTestApp"
   4. Configure Consumer Administrator/User for the consumer. For example, configure 'Admin' as the administrator for this consumer. 'Guest' as the user for this consumer.
   5. Resource groups checked should be GpuComputeHosts, if created, otherwise ComputeHosts and ManagementHosts.
   6. OS user account should be set to the OS user account the services will run under.
4. Register the test GPU application:

Note: The application profile is located at $SOAM\_HOME/conf/GpuTestApp.xml. The application service was already deployed to the compute hosts in step 2 of the installation instructions.

* 1. Change directory to where the application profile is located:

$ cd $SOAM\_HOME/conf

* 1. Register the GPU sample application profile to Symphony:

$ soamreg GpuTestApp.xml

1. Test the sample application:

The application sums two 10 element vectors containing random values 10 times

* 1. Source the Symphony environment. Default install location is /opt/ego:

$ cd /opt/ego # cd to install location

$ source cshrc.platform # csh

-or-

$ . ./profile.platform # sh

* 1. Run the sample application client:

$ GpuTestClient

For more options, run

$ GpuTestClient –h

## New metrics description

This section explains the GPU-related metrics collected and reported by Symphony.

* + ngpus

Number of GPUs detected on the host.

* + gpushared

Number of GPUs in Normal mode detected on the host.

* + gpuexclusive\_thread

Number of GPUs in Exclusive Thread mode detected on the host.

* + gpuprohibited

Number of GPUs in Prohibited mode detected on the host.

* + gpuexclusive\_process

Number of GPUs in Exclusive Process mode detected on the host (support started with CUDA 4.0).

* + gpudriverversion

CUDA driver version installed on the host (for example 260.19.26)

* + gpusdkversion

CUDA SDK version installed on the host (for example 3.2)

* + gpucap1\_0

Number of devices with capability 1.0

* + gpucap1\_1

Number of devices with capability 1.1

* + gpucap1\_2

Number of devices with capability 1.2

* + gpucap1\_3

Number of devices with capability 1.3

* + gpucap2\_plus

Number of devices with capability 2.0 or higher

For each GPU device number 'X' on the host:

* + gpucapverX

The capability of the device.

* + gpumodeX

GPU device compute mode.

* + gputempX

Temperature of the GPU device.

* + gpueccX

Number of ECC errors detected on the GPU device.

* + gpuutX

GPU utilization of the device.

* + gpumemX

Current virtual memory available.

* + gpumaxmemX

Maximum virtual memory available.

* + gpuncoresX

Number of cores per device.

## Uninstall Instructions

1. Source the Symphony environment. Default install location is /opt/ego:

$ cd /opt/ego # cd to install location

$ source cshrc.platform # csh

-or-

$ . ./profile.platform # sh

1. Log on to Symphony as Cluster Administrator:

$ egosh user logon

1. Uninstall the deployment package from the hosts:

$ rsdeploy uninstall gpupack

Get the status to check for possible errors

$ rsdeploy status gpupack

1. Restart the cluster
   1. Log on to Symphony as Cluster Administrator:

$ egosh user logon

* 1. Shut down the WEBGUI service:

$ egosh service stop WEBGUI

* 1. Restart the cluster:

$ egosh ego restart all