

# NSIGHT ECLIPSE PLUGINS EDITION

DG-06450-001 \_v9.1 | April 2018

### **Getting Started Guide**

### TABLE OF CONTENTS

Chapter 1. Introduction	1
1.1. About Nsight Eclipse Plugins Edition	. 1
Chapter 2. Using Nsight Eclipse Edition	2
2.1. Installing Nsight Eclipse Edition	2
2.1.1. Installing CUDA Toolkit	. 2
2.1.2. Configure CUDA Toolkit Path	. 2
2.2. Nsight Eclipse Main Window	. 4
2.3. Creating a New Project	5
2.4. Importing CUDA Samples	6
2.4.1. cuHook Sample	6
2.5. Configure Build Settings	7
2.6. Debugging CUDA Applications	8
2.7. Remote development of CUDA Applications	10
2.8. Debugging Remote CUDA Applications	. 12
2.9. Profiling CUDA applications	
2.10. Importing Nsight Eclipse Projects	19
2.11. More Information	

### LIST OF FIGURES

Figure 1	Nsight main window after creating a new project	5
Figure 2	Debugging CUDA application	9
Figure 3	Debugging CUDA application	10
Figure 4	Debugging remote CUDA application	18
Figure 5	Profiling CUDA Application	19

# Chapter 1. INTRODUCTION

This guide introduces Nsight Eclipse Plugins Edition and provides instructions necessary to start using this tool. Nsight Eclipse is based on Eclipse CDT project. For a detailed description of Eclipse CDT features consult the integrated help "C/C++ Development User Guide" available from inside Nsight (through Help->Help Contents menu).

### 1.1. About Nsight Eclipse Plugins Edition

NVIDIA<sup>®</sup> Nsight<sup>™</sup> Eclipse Edition is a unified CPU plus GPU integrated development environment (IDE) for developing CUDA<sup>®</sup> applications on Linux and Mac OS X for the x86, POWER and ARM platforms. It is designed to help developers on all stages of the software development process. Nsight Eclipse Plugins can be installed on vanilla Eclipse 4.4 or later using the standard Help->Install New Software.. Menu. The principal features are as follows:

- Edit, build, debug and profile CUDA-C applications
- CUDA aware source code editor syntax highlighting, code completion and inline help
- Graphical user interface for debugging heterogeneous applications
- Profiler integration Launch visual profiler as an external application with the CUDA application built in this IDE to easily identify performance bottlenecks

For more information about Eclipse Platform, visit http://eclipse.org

# Chapter 2. USING NSIGHT ECLIPSE EDITION

### 2.1. Installing Nsight Eclipse Edition

Nsight Eclipse Plugins archive is part of the CUDA Toolkit. Nsight Eclipse Plugins archive can be installed using the Help -> Install New Software... Menu on Eclipse 4.4 or later

### 2.1.1. Installing CUDA Toolkit

To install CUDA Toolkit:

- **1.** Visit the NVIDIA CUDA Zone download page: http://www.nvidia.com/object/cuda\_get.html
- **2.** Select appropriate operating system. Nsight Eclipse Edition is available in Mac OS X and Linux toolkit packages.
- 3. Download and install the CUDA Driver.
- 4. Download and install the CUDA Toolkit.
- 5. Follow instructions to configure CUDA Driver and Toolkit on your system.

### 2.1.2. Configure CUDA Toolkit Path

To get started, CUDA Toolkit path must be configured in Eclipse with Nsight Plugins:

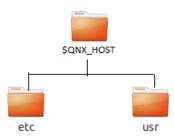
- 1. Open the Preferences page, Window > Preferences.
- 2. Go to CUDA toolkit section.
- **3.** Select the CUDA toolkit path to be used by Nsight. CUDA tookits that are installed in the default location will automatically appear.

× + Preferences					
type filter text 🛛 🖾	CUDA			(-) ▼ → ▼ ▼	
<ul><li>General</li><li>Ant</li></ul>	CUDA Toolkit	/usr/local/cuda-9.0/bin	÷	Add new SDK location	
▶ C/C++	Target OS:	Native	* *		
Code Recommenders	Target Architecture:	x86-64	* *	32 Bit	
CUDA	CCBIN	q++		Browse	
Data Management		9**	]	Drowse	
Dynamic Languages					
Help					
Install/Update					
▶ Java	CUDA Software Preer	nation Dobugging			
▶ Maven	CODA SOItware Preer	inption Debugging			
Mylyn		CUDA software preemption enables single-GPU debugging and simultaneous debugging of multiple CUDA applications on the same device. Software preemption is supported on SM 3.5+ devices and not required for SM 6.0+ devices.			
<ul> <li>Plug-in Development</li> <li>Remote Systems</li> </ul>					
<ul> <li>Report Design</li> </ul>					
<ul> <li>Ruby</li> </ul>	🗹 Enable CUDA soft	ware preemption debugging			
<ul> <li>Run/Debug</li> </ul>					
▶ Team					
Terminal					
Validation					
WindowBuilder	Make cuda-gdb and N	VIDIA visual profiler as default launchers			
▶ XML			Dechase	Dofaulte Apply	
			Restore [	Defaults Apply	
?			Can	cel OK	

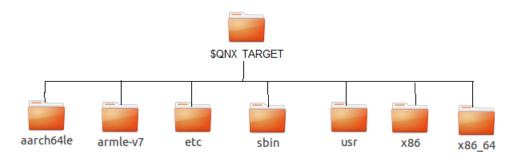
- **4.** CUDA toolkit path can be also specified in the project properties page in order to use different toolkit for a project.
- **5.** For QNX: When QNX is selected as Target OS, a dialog will be displayed to set the QNX\_HOST and QNX\_TARGET environment variables if they were not already set.

8				
Set QNX_HOST and QNX_Target Environment Variables Set Environment Variable				
QNX_HOST:	/p4/sw/tools/embedded/qnx/qnx700-ga1/host/linux/x86_64 Browse			
QNX_TARGET:	udeen/p4/sw/tools/embedded/qnx/qnx700-ga1/target/qnx7 Browse			
?	Cancel OK			

QNX\_HOST environment variable identifies the directory that holds the host-related components:



QNX\_TARGET environment variable identifies the directory that holds the targetrelated components:



## 2.2. Nsight Eclipse Main Window

On the first run Eclipse will ask to pick a workspace location. The workspace is a folder where Nsight will store its settings, local files history and caches. An empty folder should be selected to avoid overwriting existing files.

The main Nsight window will open after the workspace location is selected. The main window is divided into the following areas:

- *Editor* displays source files that are opened for editing.
- Project Explorer displays project files
- *Outline* displays structure of the source file in the current editor.
- *Problems* displays errors and warnings detected by static code analysis in IDE or by a compiler during the build.
- *Console* displays make output during the build or output from the running application.

### 2.3. Creating a New Project

- **1.** From the main menu, open the new project wizard **File** > **New...** > **C/C++ Project**
- 2. Specify the project name and project files location.
- 3. Specify the project type like executable project.
- 4. Specify the CUDA toolchain from the list of toolchains.
- 5. Specify the project configurations on the next wizard page.
- 6. Complete the wizard. The project will be shown in the **Project Explorer** view and source editor will be opened.
- 7. Build the project by clicking on the hammer button on the main toolbar.

😠 🖨 💷 C/C++ - bitreverse/src/bitreverse.cu - Nsight					
File Edit Source Refactor Navigate Search Project Run Window Help					
C1 + E E E B + S + K + E   21 + S1 C + C +   ≫ + O + 9 + 4 +   22 - 24 + [   9 + 5 + 4 + 4 +	<b>9</b> II II	E	t		
Project E ☆ 🖓 🗖 🖸 🕑 bitreverse.cu 🕸		- D 🗄 o 🛱			
<pre>* * Copyright 1993-2012 NVIDIA Corporation. All right #include <stdio.h> #include <stdio.h> #include <stdib.h> static const int WORK_SIZE = 256; * static const int WORK_SIZE = 256; * fits macro checks return value of the CUDA runtime * the application if the call failed. */ #define CUDA_CHECK_RETURN(value) {     deviceunsigned int bitreverse(unsigned int numbe     number = ((0xf0f0f0f0 &amp; number) &gt;&gt; 4)   ((0x0f0f0     number = ((0x60f0f0f0 &amp; number) &gt;&gt; 4)   ((0x0f0f0     number = ((0xaaaaaaa &amp; number) &gt;&gt; 4)   ((0x3333     number = ((0xaaaaaaa &amp; number) &gt;&gt; 4)   ((0x35555     return number; }  e/** * CUDA kernel function that reverses the order of bi */ globalvoid bitreverse(void *data) {     unsigned int *idata = (unsigned int*) data;     idata[threadIdx.x] = bitreverse(idata[threadIdx.x] */ Problems X  Tasks Console Properties Oitems Description Resource</stdib.h></stdio.h></stdio.h></pre>	call and exits r) { fof & number) << 4 333 & number) << 2 555 & number) << 1 ts in each element 1):	↓ <sup>2</sup> z × u sta u sta u <sup>s</sup> ta u <sup>s</sup> ta u ta t ta u <sup>s</sup> ta u ta t t	₹ 🖋 ● 🗮 ▽ dio.h		
□         Writable         Smart Insert         27:1					

Figure 1 Nsight main window after creating a new project

## 2.4. Importing CUDA Samples

The CUDA samples are an optional component of the CUDA Toolkit installation. Nsight provides a mechanism to import these samples and work with them easily:

Samples that use the CUDA driver API (suffixed with "Drv") are not supported by Nsight.

- **1.** From the main menu, open the new project wizard **File** > **New...** > **C/C++ Project**
- 2. Specify the project name and project files location.
- 3. Select Import CUDA Sample under Executable in the Project type tree.
- 4. Select CUDA toolchain from the Toolchains option. location.
- **5.** On the next wizard page select project sample you want to import. Also select the target CPU architecture. Press **Next...**
- 6. Specify the project parameters on the next wizard page.
- 7. Complete the wizard. The project will be shown in the **Project Explorer** view and source editor will be opened.
- 8. Build the project by clicking on the hammer button on the main toolbar.

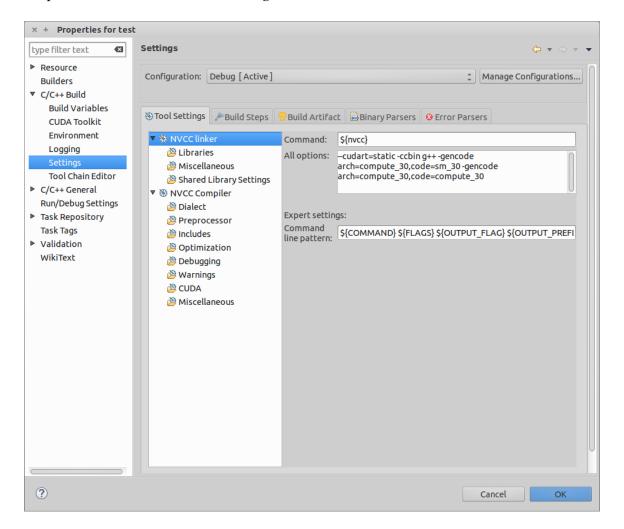
### 2.4.1. cuHook Sample

cuHook sample builds both the library and the executable. cuHook sample should be imported as the "makefile" project using the following steps.

- 1. From the main menu, open the new project wizard File > New... > C/C++ Project
- 2. Select project type "Makefile project" and choose "Empty Project"
- 3. Specify the project name and project files location.
- **4.** Complete the wizard. The project will be shown in the **Project Explorer** view.
- 5. Right click on the project Import... > General > File System
- 6. On the next wizard page, select the location of cuHook sample(Samples/7\_CUDALibraries/cuHook)
- 7. Select all the source files and makefile and Finish the wizard
- 8. Build the project by clicking on the hammer button on the main toolbar.
- 9. To run the sample, from the main menu Run > Run Configurations... > Select the executable > Go to Environment tab > New... > enter Name=LD\_PRELOAD, Value=./ libcuhook.so.1 > Run will execute the sample

### 2.5. Configure Build Settings

To define build settings: In the C/C++ Projects view, right-click your project, and select Properties. Select C/C++ Build, Settings from the list.



The following are the categories of Nvcc linker settings that can be configured for the selected project.

All options field in the main page is not editable and it's the collection of options set in the child categories.

 Libraries - Configure library search path(-L) and to include linker libraries(-l). When you are cross compiling for different target os, the library search path should point to the appropriate location where the target os libraries are present.

😣 🗊 Properties for mas	er	
type filter text 🛛 🕱	Settings 🗘 🔻 🖒 🔻 🖒	,
<ul> <li>Resource</li> <li>Build</li> <li>Build Variables</li> <li>Environment</li> <li>Logging</li> <li>Settings</li> <li>Target Systems</li> <li>Tool Chain Editor</li> <li>Builders</li> <li>C/C++ General</li> <li>Run/Debug Settings</li> </ul>	Configuration: Debug [Active] Configuration: Debug [Active] CUDA Tool Settings Build Steps Build Artifact Binary Parsers CUDA Error Parsers CUDA Runtime Library: Static CUDA Runtime Library: Static CUDA Runtime Library: Static Libraries (-1) CUDA Runtime Library: Static Libraries (-1) CUDA Runtime Library: Static Libraries (-1) CUDA Runtime Library: Static CUDA Runtime Library	
(())))	<sup>™</sup> Libraries <sup>№</sup> Miscellaneous <sup>™</sup> Miscellaneous <sup>№</sup> Nybuf_utils <sup>™</sup> Nybuf_utils	
?	Cancel	

- Miscellaneous Set additional linker options and option to link with OpenGL libraries.
- *Shared Library Settings* Set option to build a shared library.

The following are the categories of Nvcc Compiler settings that can be configured for the selected project.

All options field in the main page is not editable and it's the collection of options set in the child categories.

- *Dialect* Select the language standard and dialect options.
- Preprocessor Add the defined and undefined symbols for the preprocessor.
- *Includes* Set include paths and include files for the compiler.
- *Optimization* Set the compiler optimization level.
- *Debugging* Set the options to generate debug information.
- *Warnings* Set inhibit all warning messages.
- CUDA Generate code for different real architectures with the PTX for the same vitrual architectures.

## 2.6. Debugging CUDA Applications

Nsight must be running and at least one project must exist.

- **1.** In the **Project Explorer** view, select project you want to debug. Make sure the project executable is compiled and no error markers are shown on the project.
- **2.** Right click on the project and go to **Debug As > NVIDIA CUDA GDB Debugger** menu.
- **3.** You will be offered to switch perspective when you run debugger for the first time. Click "Yes".

Perspective is a window layout preset specifically designed for a particular task.

- **4.** Application will suspend in the *main* function. At this point there is no GPU code running.
- 5. Add a breakpoint in the device code. Resume the application.

Debugger will break when application reaches the breakpoint. You can now explore your CUDA device state, step through your GPU code or resume the application.

🛞 🖨 🗈 Debug - blackschole/BlackScholes_kernel.cuh - Eclipse Platform	
1 • 11 • 11 10 📾 🗉 🔌 🕨 🛛 🖷 11 3. 19 12 12 12 12 12 12 12 12 12 12 12 12 12	\$\$ • <b>0</b> • <b>9</b> • <b>∅ ∅ ∅ ∂</b> • <b>√</b> • <b>√</b> • <b>√</b> • <b>√</b> • <b>√</b> • <b>√</b> •
	Quick Access 🖻 🖻 Resource 🗟 C/C++ 🗱 Debug
🎋 Debug 🛿 🦂 🦗 🔐 🗢 🖘 🗢 🗖	(*)= Variables 🛛 💁 Breakpoints 🍕 Expressions
▼ ⓒ blackschole [C/C++ Application]	
▼	Name Type T(0,0,0)B(0,0,0)
▼ % CUDA Thread (0,0,0) Block (0,0,0)	▶
BlackScholesGPU() at BlackScholes_kernel.cuh:90 0x8fff60	▶
▶ SAll Kernel Threads (15,625 Blocks of 128 Threads)	▶
▶ 🔐 Host Process [blackschole] [5357] [cores: 1,3]	had OptionStella costrict @ opporte flast + costrict @ OvEDE100000
Mill GOD	
<pre>   BlackScholes.cu  BlackScholes_gold.cpp</pre>	
🖵 Console 🛱 🗟 Tasks 📳 Problems 🕐 Executables	
blackschole [C/C++ Application] blackschole	
Initializing data allocating CPU memory for options. allocating GPU memory for options. generating input data in CPU mem. copying input data to GPU mem. Data init done.	

#### Figure 2 Debugging CUDA application

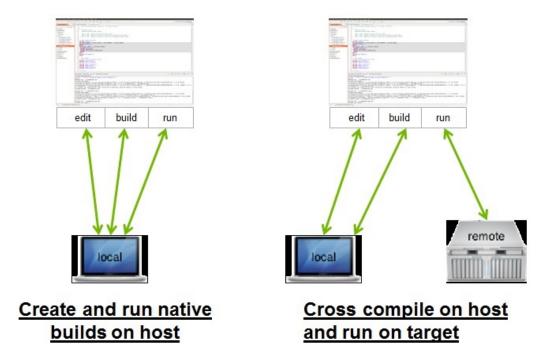
Additional debugger options can be set in the debug configuration dialog through Run > Debug Configurations .. menu..

😣 🗊 Debug Configurations					
Create, manage, and run configuration	S		- A		
Image: Second Secon	<ul> <li>Extrapolate "optimized out</li> <li>Force thread list update on a</li> <li>CUDA GDB executable:</li> <li>CUDA GDB init file:</li> </ul>	re kernel is launched or terminated " CUDA register values	Browse		
Filter matched 17 of 27 items	Using CUDA GDB Create Process L	auncher - Select other	vert Apply		
?		C	close Debug		

Figure 3 Debugging CUDA application

### 2.7. Remote development of CUDA Applications

Nsight Eclipse Edition also supports remote development of CUDA application starting with CUDA Toolkit 6.0. The picture below shows how Nsight Eclipse Edition can be used for local as well as remote development:



For remote development you do not need any NVIDIA GPU on your host system. The remote target system can be a Linux x86 or POWER system with an NVIDIA GPU or an Tegra-based ARM system. Nsight IDE and UI tools can only be hosted on x86 and POWER systems.

Nsight Eclipse Plugins supports the cross compilation mode for remote devices.

In the **cross compilation mode** the project resides on the host system and the cross compilation is also done on the host system. The cross compilation mode is only supported on an Ubuntu x86 host system.

To cross compile select the target cross compile architecture in CPU architecture drop down in the project properties page:

× + Properties for BlackScholes				
type filter text 🛛 🖾	CUDA Toolkit			↓ ↓ ↓ ↓ ↓
Resource Builders	CUDA Toolkit	/usr/local/cuda-9.0/bin	÷	Add new SDK location
▼ C/C++ Build Build Variables	Target OS: Target Architecture:	Native           ARM		✓ 32 Bit
CUDA Toolkit Environment Logging Settings Tool Chain Editor C/C++ General Run/Debug Settings Task Repository Task Tags Validation WikiText	CCBIN	arm-linux-gnueabihf-g++		Browse
			Restore D	efaults Apply
?			Cance	el OK

### 2.8. Debugging Remote CUDA Applications

Remote debugging is available starting with CUDA Toolkit 5.5. A dedicated GPU is not required to use Nsight remote debugging UI. A dedicated GPU is still required on the debug target. Only Linux targets are supported. Debug host and target may run different operating systems or have different CPU architectures. The remote machine must be accessible via SSH and CUDA Toolkit must be installed on both machines.

If there is a firewall between the host and the target, it must be set up to let RSP messages through, or SSH port-forwarding must be used.

 Select the project and right click then go to Debug As...>NVIDIA CUDA GDB Debugger(Remote) menu item.

Debug As	•	<u>1</u> Local C/C++ Application
Profile As	Þ	2 NVIDIA CUDA GDB Debugger
Compare With	•	3 NVIDIA CUDA GDB Debugger (Remote)
Restore from Local History		De <u>b</u> ug Configurations

**2.** Type the full path to a local executable or select one using the **Local file...** button.

😣 🕕 Debug CUDA /	pplication			
Debug application	n a remote sys	tem		
Run cuda-gdbserver r	motely and conn	ect debugger		
O Upload local exec	itable 🔿 De	bug remote executable	e 🔿 Do not transfer	executable
Local Executable	/blackschole/De	bug/blackschole		
			Workspace	Local File
Remote connection:	ubuntu@172.17	.162.109	•	Manage
Remote executable:	/tmp/cuda-debu	ıg/blackschole		Browse
Program arguments:				
cuda-gdbserver port:	2345 - +	)		
i ubuntu@172.17.10	2.109 is not conn	ected. Remote paths wi	ill not be validated.	Connect
?	<	Back Next >	Cancel	Finish

- **3.** Select a remote connection from a drop-down list or press the **Add connection**... button to create a new one.
- **4.** If you are creating a new remote connection, select the **SSH Only** connection type, press **Next**, and type the host name(or IP address) as well as the connection name and description (both are optional) and then press **Finish**.

😣 🗊 New R	emote Connection
Remote Con	nections
Manage avail	able connections
ubuntu@17	72.17.162.232
Host name:	172.17.162.232 Remove
User name:	ubuntu
Label:	ubuntu@172.17.162.232
System type:	SSH v Port number: 22 - +
?	Cancel Finish

**5. For Android devices:** To configure the remote connection using Android debug bridge, select the **Android debug bridge** from the Remote Connection drop-down list, Android device must be connected to the host system using USB port.

× + Debug CUDA Application					
Debug application	on a remote system				
Run cuda-gdbserver r	emotely and connect debugger				
Upload local executable					
Local Executable	/async/Debug/async				
		Workspace Local File			
Remote connection:	Android debug bridge	▼ Manage			
Remote executable:	/data/local/tmp/async	Browse			
Program arguments:					
cuda-gdbserver port:	2345 - +				
i Android debug bridge is not connected. Remote paths will not be validated. Connect					
?	< Back Next >	Cancel Finish			

Press **Manage** button, and enter or select the path to adb utility. You need to install Android SDK platform tools to use Android debug bridge. press **Detect** button to find the android device available through ADB.

×					
ADB Connection					
Configure A	DB Connection				
ADB path:	deen/JetPack_Android2/android-sdk-linux/platform-tools/adb	Browse			
ADB devices:	P2180A00P03600000505	Detect			
?	Cancel	OK			

- 6. Optional: Press **Connect** to verify the selected remote connection.
- 7. Press the Next button.
- **8.** Type the full path to cuda-gdbserver on the remote system or select one using the **Browse...** button.

😣 🗊 Debug	UDA Application			
Configure CU	DA Toolkit			
Configure CUD	A toolkit location on the remo	te system		
Toolkit path:	/usr/local/cuda/bin		Browse	Detect
Library paths:	🗁 /usr/local/cuda/lib64			Browse
	/usr/local/cuda/lib			Delete
	🐈 Add new path			Ŷ
				₽.
i ubuntu@1	2 17 162 100 is pat copported	Pomoto paths will pot ho vali	dated	Connect
1 ubuntu@1	2.17.102.10915 not connected	. Remote paths will not be vali	dated.	Connect
?	< Back	Next > Canc	el	Finish

- **9.** Click on "Add new path" or on the **Browse...** button to specify the path to the shared libraries the remote application depends on.
- **10**Click on the **Finish** button to finish the new debug configuration wizard and start debugging the application.
- **11**.You will be offered to switch perspective when you run the debugger for the first time. Click **Yes**.

Perspective is a window layout preset specifically designed for a particular task.

The debugger will stop at the application main routine. You can now set breakpoints, or resume the application.

😕 💿 💿 Debug - /home/test/bitreverse/bitreverse.cu - Nsight				
<b>○ · □ ● ○ ◆ · ○ · ○ · ○ · ○ · ○ </b>   <b>●</b> □ ● □ <b>●</b> 	3. 🧿 🧟 🖮 🧮 🕱 🎽 🥔 🖋	▼ Ē v "		
🏂 Debug 🛛 👷 📓 👔 🕴 🍟 🍟 🗖	🕬= Variable 🛱 💊 Breakpo 🜊 Cl	JDA 🚟 Register 🛋 Modules 🛛 🗖 🗖		
<ul> <li>▼ E bitreverse on debugger-target [C/C++ Remote Application]</li> <li>▼ B Host Process [bitreverse] [cores: 0]</li> <li>▼ P Thread [1] 10000 [core: 0] (Suspended : Breakpoint)</li> <li>≡ main() at bitreverse.cu:40 0x4027d0</li> </ul>	Name Type	via ⊡ S × % ⊡ ⊡ ▼ Value		
■ main() at bit everse.td:40 0x402700 ▶ Remote Shell ▶ gdb				
الله bitreverse.cu ک		🗄 Outline 🕱 🗖 🗖		
<pre>* Host function that prepares data array and passes it to */ int main(void) {     bitreverse&lt;&lt;&lt;1, WORK_SIZE, WORK_SIZE * sizeof(int)&gt;&gt;&gt;     CUDA_CHECK_RETURN(cudaThreadSynchronize()); // Wait for     CUDA_CHECK_RETURN(cudaGetLastError());     CUDA_CHECK_RETURN(cudaDeviceReset());     return 0; }</pre>	0:			
Console 🛛 🖉 Tasks 🗷 Problems 🛈 Executables 🚺 Memory	🔳 🗙 💥 🗎 🖬 🚛			
bitreverse on debugger-target [C/C++ Remote Application] gdb Coalescing of the CUDA commands output is off.	1			
□□				

Figure 4 Debugging remote CUDA application

## 2.9. Profiling CUDA applications

Nsight must be running and at least one project must exist. Profiler cannot be used when debugging session is in progress.

Nsight Eclipse Edition profiling features are based on the NVIDIA Visual Profiler (*nvvp*) code. Nsight Eclipse Plugins Edition will launch the Visual Profiler as an external tool with the executable and other information from the selected project.

- **1.** In the **Project Explorer** view, select project you want to profile. Make sure the project executable is compiled and no error markers are shown on the project.
- 2. Select the project and right click and go to Profile As>NVIDIA Visual Profiler menu.

Profile As	<u>1</u> Local C/C++ Application
Compare With	<u>2</u> NVIDIA Visual Profiler
Restore from Local History	<u>3</u> NVIDIA Visual Profiler (Remote)
Run C/C++ Code Analysis	Profile Configurations

Nsight Eclipse will launch the Visual Profiler to specify extra profiler options with the executable information already passed from the selected project.

File View Window Help	😣 🗊 Create New	Session			
	Executable Prope	rties			- 0
	Set executable prop	Set executable properties			
	Connection:	Local Cocal	nections		
	Toolkit:	CUDA Toolkit 8.0 (/usr/local/cuda-8.0/bin/)	Manage		
	File:	/home/kjalaludeen/runtime-NsightPlugins3/blackschole/Debug/blackschol	Browse		
	Working directory:	[Enter working directory [optional]	Browse		
	Arguments:	Enter command-line arguments			
		Profile child processes			
	Environment:	Name Value	Add		
			Delete		
🖿 CPU Detail 🖾 😑 🗖				7 - 0	🗆 Pr 🖾 🗝 🗖
1. 1. 5					
(Thread) 🗵		< Back Next > Cancel	Finish		Select or highlight a single interval to
Event			_		see properties

Figure 5 Profiling CUDA Application

### 2.10. Importing Nsight Eclipse Projects

The projects that are created with Nsight Eclipse Edition can be imported into the Eclipse workbench with Nsight Eclipse plugins.

- **1.** Open Nsight Eclipse edition and select the project that needs to be exported.
- 2. Right click on the Nsight Eclipse project and go to Export > C/C++ > C/C++ Project Settings > Next menu.

× + Export				
Select				ß
Select an export de	stination:			
type filter text				×
▶ 🗁 General ▼ 🗁 C/C++				
🕆 C/C++ Proje				
📮 Team Share	d Index			
🕨 🗁 Install				
?	< Back	Next >	Cancel	Finish

- **3.** Select the project and settings to export.
- 4. Specify the "Export to file" location.
- 5. Settings will be stored in the given XML file.
- 6. Go to Eclipse workbench where the project settings needs to be imported.
- 7. Create a C/C++ Project from the main menu File > New > C/C++ Project
- 8. Specify the project name and choose Empty project type with CUDA toolchains.
- **9.** Right click on the project to import the source files. **Import > General > File System** >(From directory) or copy the source files from the existing project.
- **10I**mport the project settings like include paths and symbols using the following right click menu **Import** > C/C++ > C/C++ Project Settings >Next...
- **11**Select the location of the project settigns file and select the project and configuration on the next wizard page.

× + Import		
Import		
Import C/C++ project settings		
Settings file		
/tmp/settings.xml		Browse
Select Project	Select Configuration	
😂 newp	🕥 🛞 Debug (Active)	
<ul> <li>Ssl</li> <li>Stl</li> <li>twd</li> <li>wds</li> </ul>	Release	
Select settings to import		
🗹 🕒 Include Paths		0
Select All Deselect All		
? < Bac	ck Next > Cancel	Finish

**12**Complete the wizard.

The project settings will be imported from the file exported from Nsight Eclipse Edition.

13Build the project by clicking on the hammer button on the main toolbar.

### 2.11. More Information

More information about the Eclipse CDT features and other topics is available in the Help contents. To access Help contents select *Help->Help Contents* from the Nsight main menu.

More information about CUDA, CUDA Toolkit and other tools is available on CUDA web page at http://developer.nvidia.com/cuda

#### Notice

ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE.

Information furnished is believed to be accurate and reliable. However, NVIDIA Corporation assumes no responsibility for the consequences of use of such information or for any infringement of patents or other rights of third parties that may result from its use. No license is granted by implication of otherwise under any patent rights of NVIDIA Corporation. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all other information previously supplied. NVIDIA Corporation products are not authorized as critical components in life support devices or systems without express written approval of NVIDIA Corporation.

#### Trademarks

NVIDIA and the NVIDIA logo are trademarks or registered trademarks of NVIDIA Corporation in the U.S. and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

#### Copyright

© 2007-2018 NVIDIA Corporation. All rights reserved.

