DevKit/TestKit Setup Guide

PlayStation®Vita Programmer Tool Runtime Library Release 3.50

Table of Contents

1 Overview

Scope of This Document

This document provides an outline of the setup procedures and usage method of the development environment using Development Kit (DevKit) and Testing kit (TestKit). The main contents are as follows:

- The procedures to install development software (compilers, libraries, sample programs, etc.) on a development host computer (Windows PC)
- A summary of the procedure to connect DevKits to a development host computer
- How to update DevKits and TestKits
- Compiling and executing sample programs
- Troubleshooting
- Tutorials for application creation/debugging/performance analysis
- How to set up the development environment for PlayStation®TV

Development Environment

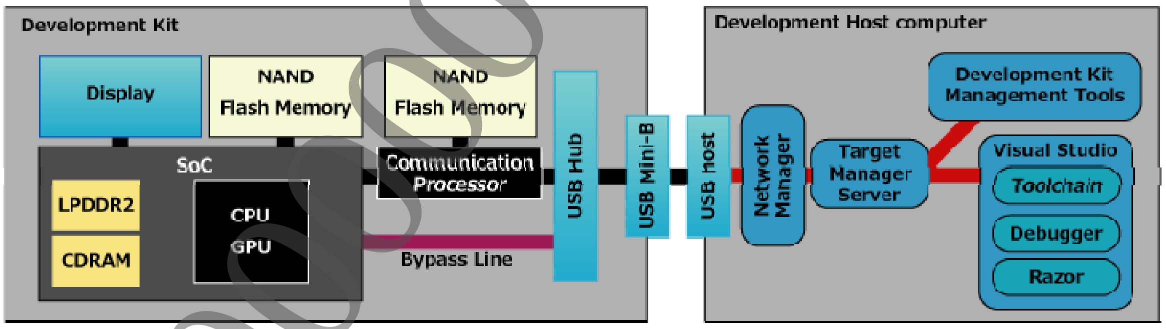
The following devices are required to develop applications for PlayStation®Vita

- Development host computer (Windows PC)
- DevKit: PDEL-1000 series
- TestKit: PTEL-1000 series or PTEL-2000 series

DevKit: PDEL-1000 Series

The system configuration is shown in the following diagram:

Figure 1 System Configuration



DevKit is device for evaluating application development; it is composed of a system equivalent to the PlayStation®Vita retail unit (PCH-1000 series) and a Communication Processor (CP) that manages communication with the development host computer. The USB port (mini-B) for development is used for connection with the development host computer via the CP.

TestKit: PTEL-1000 Series

TestKit (PTEL-1000 series) is a testing machine for title QA built based on the PlayStation®Vita retail unit (PCH-1000 series).

Note that connection with the development host computer, update, and some other methods are different from those of DevKit because the CP does not exist on the TestKit. The multi-use port is used for connection with the development host computer. For the overview of setup methods, refer to the "Appendix D: TestKit Setup" chapter.

TestKit: PTEL-2000 Series

TestKit (PTEL-2000 series) is a testing machine for title QA built based on the PlayStation®Vita retail unit (PCH-2000 series).

Note that connection with the development host computer, update, and some other methods are different from those of DevKit because the CP does not exist on the TestKit. The USB port is used for connection with the development host computer. Moreover, the provided USB Ethernet adaptor enables the setup of a test environment that uses a wired network connection. For the overview of setup methods, refer to the "Appendix D: TestKit Setup" chapter.

Development Host Computer

Perform program coding, compiling, and debugging on a separately provided development host computer. A general PC can be used. Regarding system requirements, refer to the Getting Started Guide (https://psvita.scedev.net/docs/startup/) on the PlayStation®Vita Developer Network (https://psvita.scedev.net/) (hereafter PlayStation®Vita DevNet).

SDK Overview

Compilers, libraries, sample programs, SDK Manager, and other tools are provided via the PlayStation®Vita DevNet as software to use for development. Use the SDK Manager to install them. For details on usable development software, refer to the "Programming Startup Guide" document.

2 Installing Development Software

This chapter explains how to install development software on the development host computer. SDK Manager features as a downloader, installer, SDK management and offline installer application which are combined for the SDK components.

Note
Because the USB driver is downloaded by the SDK Manager, set up the SDK on the development host computer before connecting to DevKit or TestKit.

Installation of SDK Manager

If the latest version of SDK Manager is released, SDK manager must be installed or updated. If a new SDK is released or an SDK component is to be additionally installed, it is not necessary to install SDK Manager.

(1) Log on to Windows by using a user account with administrator privileges.

(2) If Visual Studio Integration is to be used, install Visual Studio beforehand and run it at least once.

(3) Shut down any running applications that might be updated before starting installer.

(4) SDK Manager can be installed in one of two ways, as follows.

- To install PlayStation®Vita SDK for the first time, access PlayStation®Vita DevNet, download SDK_Manager-<VERSION>.exe and launch it to set up the SDK.
- If the offline installer created by SDK Manager is to be used, the application file of SDK Manager is included in the offline distribution kit. Refer to the "Appendix A: Using an offline installer" chapter.

(5) After installation is completed, SDK Manager starts automatically.

Note
To use the SDK Manager, Microsoft .NET Framework 3.5 Service Pack 1 and Windows Installer 4.5 Redistributable must be installed first.
On running SDK_Manager.exe, application files will be extracted or updated in ProgramData
`%ProgramData%\SCE\SDK Installer\Binaries\Bin`

Installation of SDK with SDK Manager

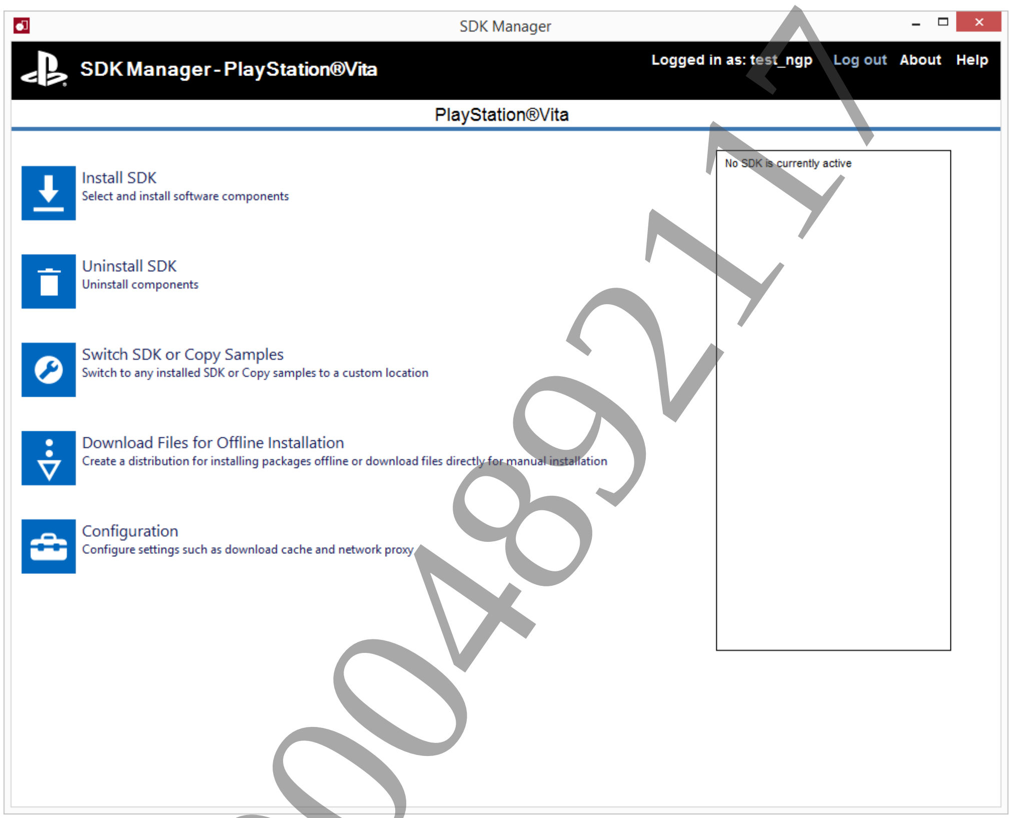
Upon SDK Manager startup, the usable contents are displayed.

The installation of new components, uninstall of existing components, active SDK switching, etc., can be used.

SDK Manager can be started from the Start menu shortcut below.

- **Start** > **All Programs** > **SCE - Common** > **SDK Manager**
  Upon SDK Manager startup, the following homepage is displayed.

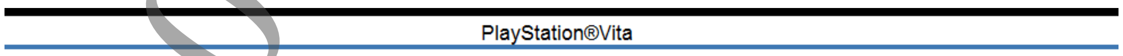
Figure 2 SDK Manager - Home Page


SDK Manager
SDK Manager - PlayStation®Vita
Logged in as: test_ngp
Log out
About
Help
PlayStation®Vita
Install SDK
Select and install software components
Uninstall SDK
Uninstall components
Switch SDK or Copy Samples
Switch to any installed SDK or Copy samples to a custom location
Download Files for Offline Installation
Create a distribution for installing packages offline or download files directly for manual installation
Configuration
Configure settings such as download cache and network proxy
No SDK is currently active

The various features of SDK Manager are as follows.

- **Platform Selection**
  Select a platform here if using a DevNet account that can access multiple platforms.

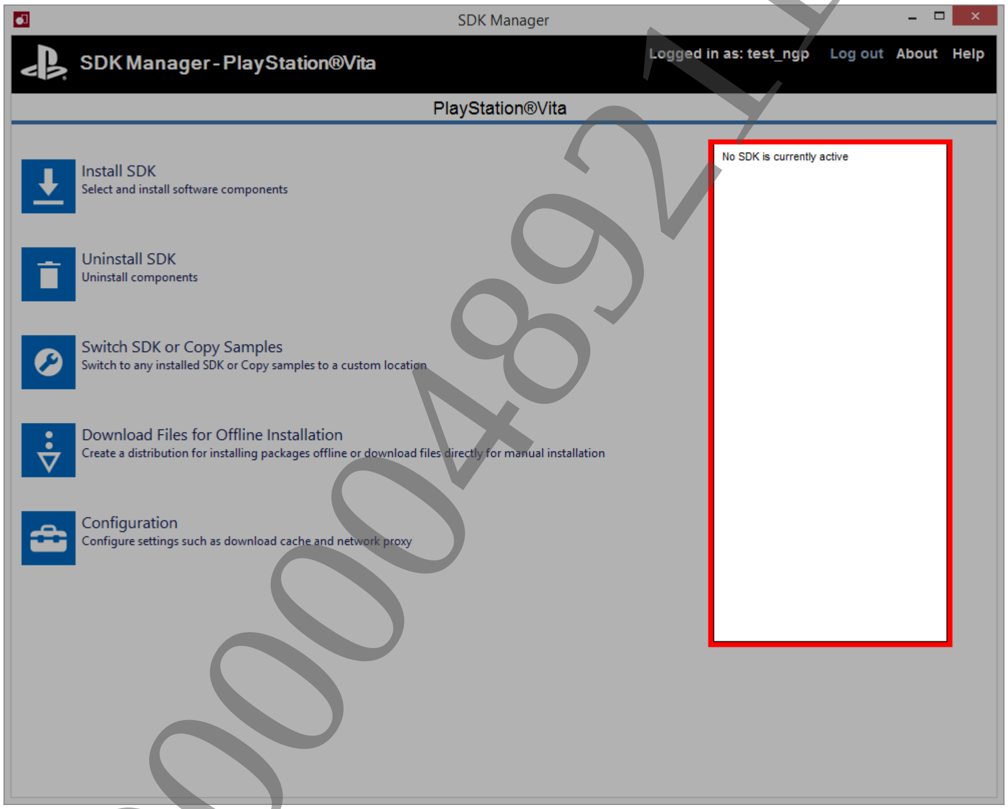
Figure 3 SDK Manager - Platform Selection



- **Install SDK**
  SDK components can be checked, selected, and installed online here.
- **Uninstall SDK**
  Installed packages can be uninstalled here.
- **Switch SDK or Copy Samples**
  Installed SDK versions can be switched and SDK sample programs can be copied to arbitrary directories.

- **Download Files for Offline Installation**
  Offline SDK distribution kits in different formats can be created here.
- **Configuration**
  Various settings (for example, download cache directory or network proxy specification) can be made here.
- **About**
  Displays the version and license information of SDK Manager.
- **Help**
  Displays the SDK Manager Help
- **SDK Information Panel**
  When an SDK is installed, environment variable `%SCE_PSP2_SDK_DIR%` is set and is managed on SDK Manager. The current SDK version and environment variable `%SCE_PSP2_SDK_DIR%` can be checked with this item.

Figure 4 SDK Manager - SDK Information Panel


SDK Manager
SDK Manager - PlayStation®Vita
Logged in as: test_ngp
Log out
About
Help
PlayStation®Vita
Install SDK
Select and install software components
Uninstall SDK
Uninstall components
Switch SDK or Copy Samples
Switch to any installed SDK or Copy samples to a custom location
Download Files for Offline Installation
Create a distribution for installing packages offline or download files directly for manual installation
Configuration
Configure settings such as download cache and network proxy

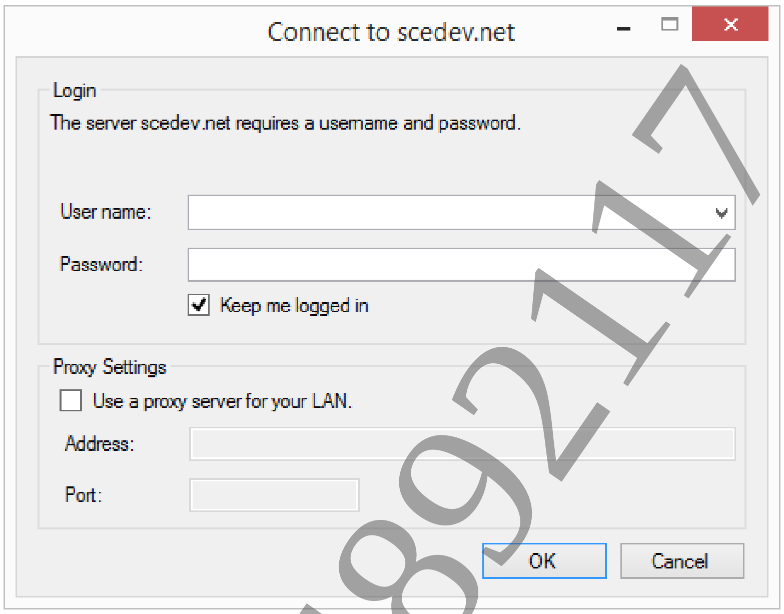
No SDK is currently active

The procedure for installing an SDK is as follows.

(1) Click the **Install SDK** button.

(2) Log in to PlayStation®Vita Developer Network.
SDK Manager accesses PlayStation®Vita DevNet to download installation components.
The login dialog is displayed, so enter the user name and password for accessing PlayStation®Vita DevNet.
If proxy server settings are required, make the settings

Figure 5 SDK Manager – Login Dialog



Connect to scedev.net

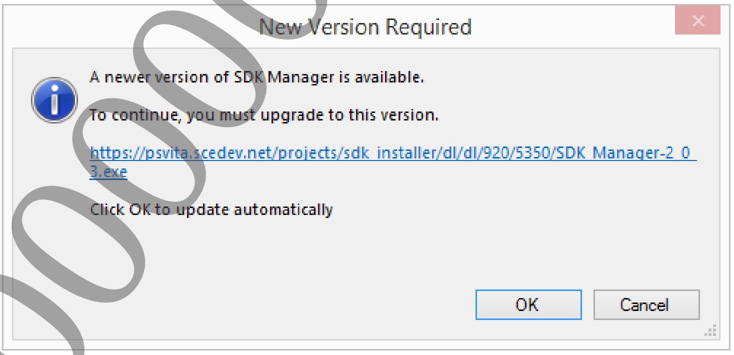
Login
The server scedev.net requires a username and password.

User name:
Password:
[✓] Keep me logged in

Proxy Settings
[ ] Use a proxy server for your LAN.
Address:
Port:

If a more recent version of SDK Manager has been released, the following update dialog is displayed after login, so click **OK** and update SDK Manager.

Figure 6 SDK Manager - Update Dialog



New Version Required

A newer version of SDK Manager is available.

To continue, you must upgrade to this version.

https://psvita.scedev.net/projects/sdk_installer/dl/dl/920/5350/SDK_Manager-2_0_3.exe

Click OK to update automatically

Note
If the **Keep me logged in** is selected in the login dialog, SDK Manager records the login information and login is executed automatically the next time it starts up. Caution is required regarding this feature if the development host computer is shared, etc. To avoid automatic login, click the **Log out** button on the top right corner of the SDK Manager homepage, and finish by logging out.

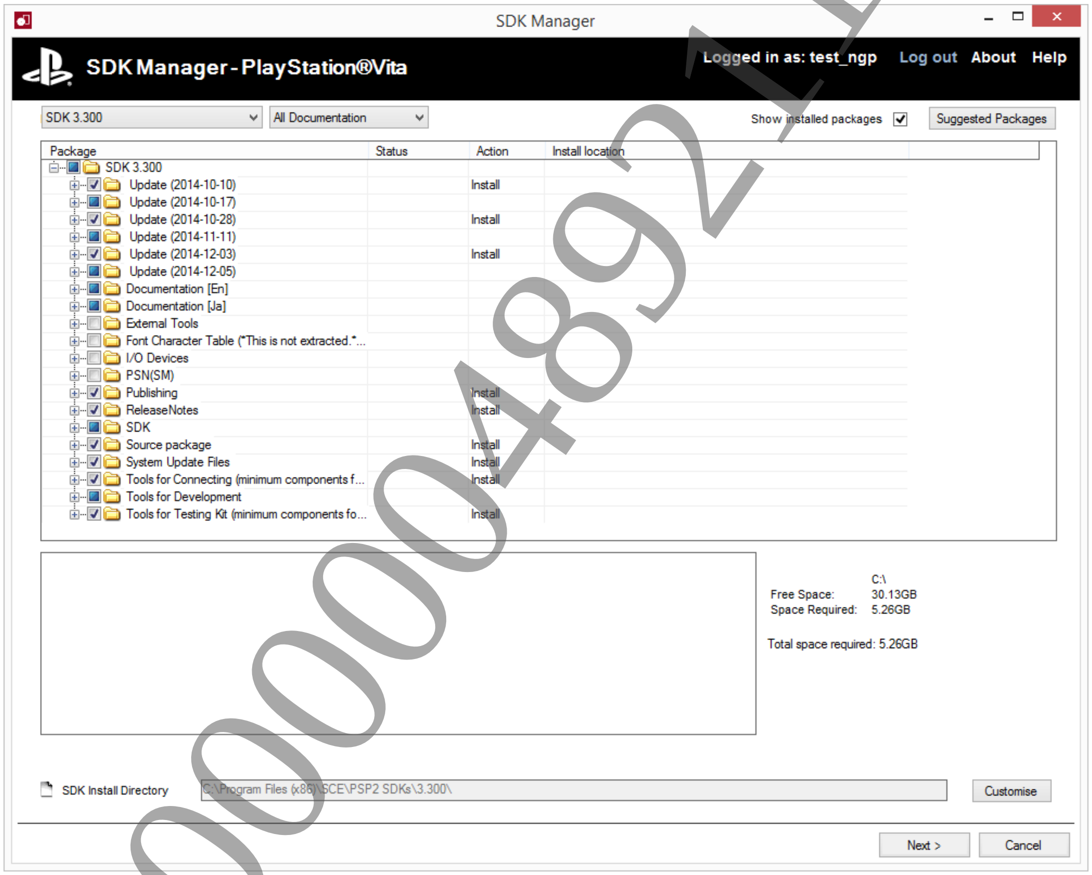
(3) Select a package to install.
The components that can be installed are displayed.

- The **SDK Version Filter** combo box at the top of the page indicates which SDK the component list is for.
  To check the components that can be used for a different SDK version, select an **SDK Version Filter** combo box option to switch the SDK version. Components that are already installed in a different directory can be checked by switching the **SDK Install Directory** at the bottom of the page.
- The suggested package configuration can be selected by selecting **Suggested Packages**.

Note
When the checkbox of **Show installed packages** is unmarked, only the packages that have been additionally released after the previous install will be displayed.

At this point, install the suggested package configuration. Click **Suggested Packages** and then click **Next**.

Figure 7 SDK Manager – Package Selection



Package	Status	Action	Install location
■ SDK 3.300			
✓ Update (2014-10-10)		Install	
■ Update (2014-10-17)			
✓ Update (2014-10-28)		Install	
■ Update (2014-11-11)			
✓ Update (2014-12-03)		Install	
■ Update (2014-12-05)			
■ Documentation [En]			
■ Documentation [Ja]			
☐ External Tools			
☐ Font Character Table (*This is not extracted.*...			
☐ I/O Devices			
☐ PSN(SM)			
✓ Publishing		Install	
✓ ReleaseNotes		Install	
■ SDK			
✓ Source package		Install	
✓ System Update Files		Install	
✓ Tools for Connecting (minimum components f...		Install	
■ Tools for Development			
✓ Tools for Testing Kit (minimum components fo...		Install	

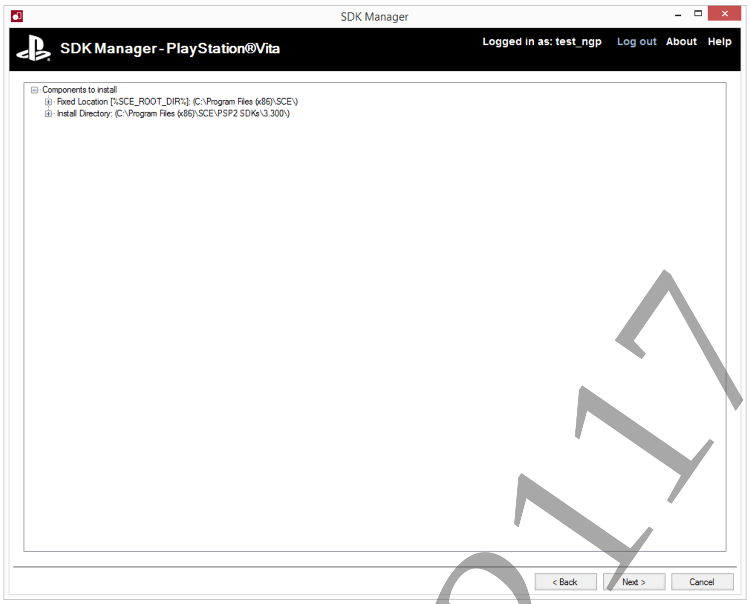
	C:\
Free Space:	30.13GB
Space Required:	5.26GB

Total space required: 5.26GB

SDK Install Directory: C:\Program Files (x86)\SCE\PSP2 SDKs\3.300\

(4) A summary in tree form is displayed for each install directory.
Verify the correct components will be installed, and then click **Next**.

Figure 8 SDK Manager - Component List

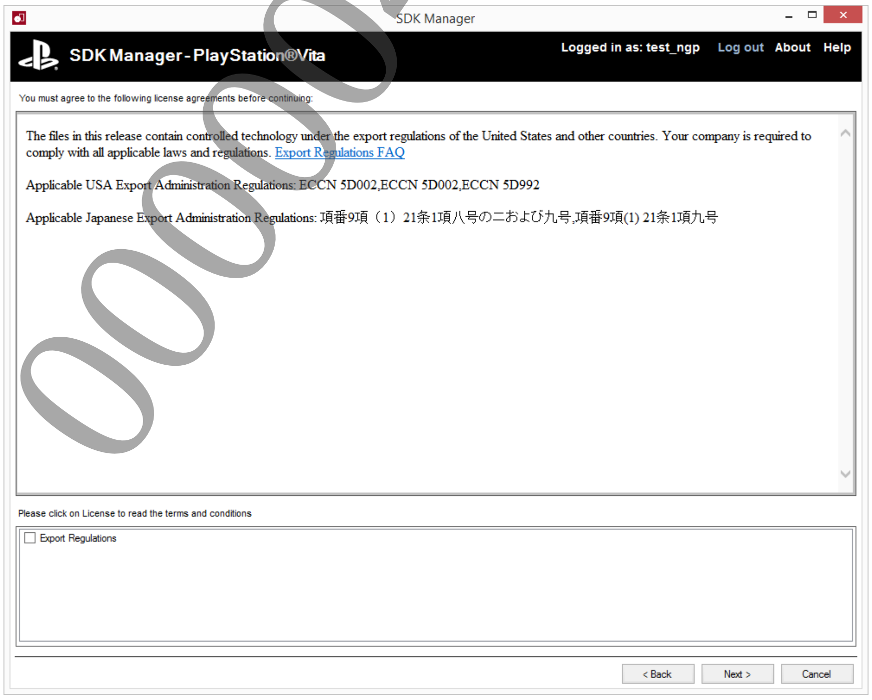


Components to install
Fixed Location [%SCE_ROOT_DIR%]: (C:\Program Files (x86)\SCE\)
Install Directory: (C:\Program Files (x86)\SCE\PSP2 SDKs\3.300\)

(5) Start downloading and installation.
When the license agreement page in Figure 9 is displayed, confirm the license/export regulations content, check the checkbox, then click the **Next** button to start downloading and installation.

Note
Downloading some components requires compliance with the laws and regulations regarding the export regulations of the U.S.A. and other countries.

Figure 9 SDK Manager - License Agreement



You must agree to the following license agreements before continuing:

The files in this release contain controlled technology under the export regulations of the United States and other countries. Your company is required to comply with all applicable laws and regulations. Export Regulations FAQ

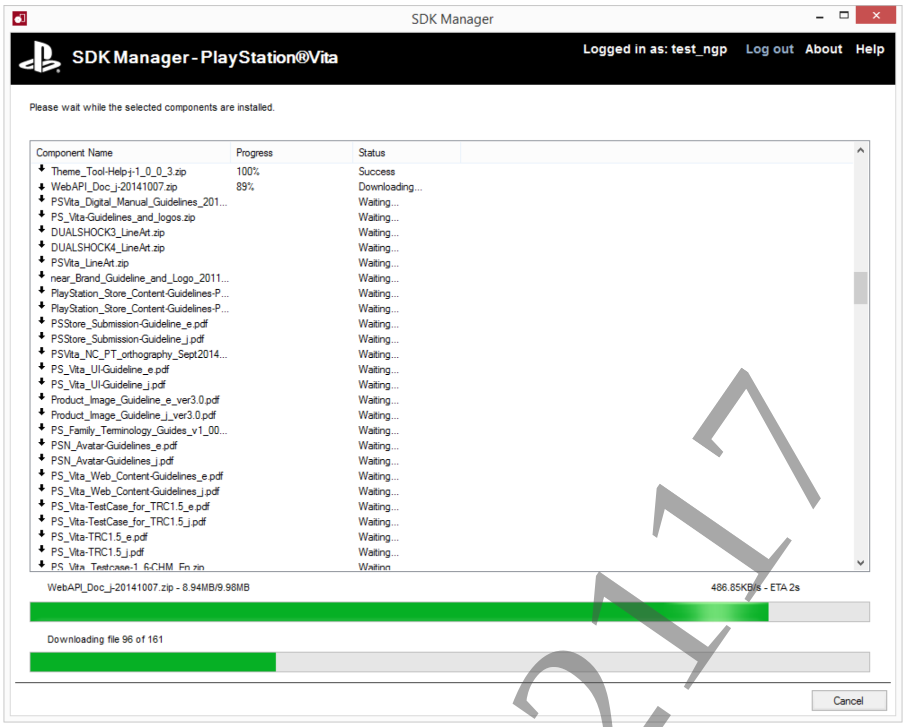
Applicable USA Export Administration Regulations: ECCN 5D002,ECCN 5D002,ECCN 5D992

Applicable Japanese Export Administration Regulations: 項番9項（1）21条1項八号の二および九号,項番9項(1) 21条1項九号

Please click on License to read the terms and conditions

☐ Export Regulations

Figure 10 SDK Manager - Download & Install


SDK Manager

SDK Manager - PlayStation®Vita

Logged in as: test_ngp Log out About Help

Please wait while the selected components are installed.

Component Name	Progress	Status
Theme_Tool-Help-j-1_0_0_3.zip	100%	Success
WebAPI_Doc_j-20141007.zip	89%	Downloading...
PSVita_Digital_Manual_Guidelines_201...		Waiting...
PS_Vita-Guidelines_and_logos.zip		Waiting...
DUALSHOCK3_LineArt.zip		Waiting...
DUALSHOCK4_LineArt.zip		Waiting...
PSVita_LineArt.zip		Waiting...
near_Brand_Guideline_and_Logo_2011...		Waiting...
PlayStation_Store_Content-Guidelines-P...		Waiting...
PlayStation_Store_Content-Guidelines-P...		Waiting...
PSStore_Submission-Guideline_e.pdf		Waiting...
PSStore_Submission-Guideline_j.pdf		Waiting...
PSVita_NC_PT_orthography_Sept2014...		Waiting...
PS_Vita_UI-Guideline_e.pdf		Waiting...
PS_Vita_UI-Guideline_j.pdf		Waiting...
Product_Image_Guideline_e_ver3.0.pdf		Waiting...
Product_Image_Guideline_j_ver3.0.pdf		Waiting...
PS_Family_Terminology_Guides_v1_00...		Waiting...
PSN_Avatar-Guidelines_e.pdf		Waiting...
PSN_Avatar-Guidelines_j.pdf		Waiting...
PS_Vita_Web_Content-Guidelines_e.pdf		Waiting...
PS_Vita_Web_Content-Guidelines_j.pdf		Waiting...
PS_Vita-TestCase_for_TRC1.5_e.pdf		Waiting...
PS_Vita-TestCase_for_TRC1.5_j.pdf		Waiting...
PS_Vita-TRC1.5_e.pdf		Waiting...
PS_Vita-TRC1.5_j.pdf		Waiting...
PS_Vita_Testcase-1_6-CHM_En.zip		Waiting

WebAPI_Doc_j-20141007.zip - 8.94MB/9.98MB 486.85KB/s - ETA 2s

Downloading file 96 of 161

(6) When the installation completes, the Terms of Service are displayed. Click **Next** to display the Installation completed message.
To return to the homepage click **Finish**. This completes the installation. When the reboot of a computer is required after installation, reboot your computer. The selected components are installed to installation directory. Refer to the "Verifying the Setup" section about the details of the installation contents.

Other SDK Manager Features

Configuring a Proxy Server

In an environment where a connection is made to the Internet via a proxy server, proxy server configurations must also be made to the SDK Manager. Make settings as follows.

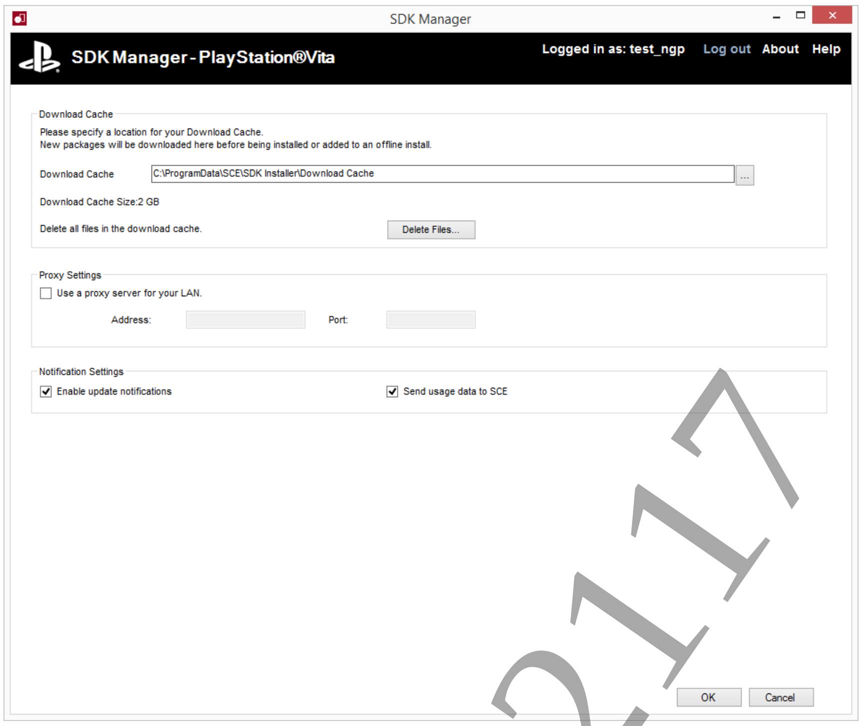
(1) Click the **Configuration** button on the homepage.

(2) In the Proxy Settings section, select the **Use a proxy server for your LAN** check box and enter the address and port of the proxy server. From the **Proxy Settings** section, mark the **Use a proxy server for your LAN.** checkbox and enter the **Address** and **Port** of the proxy server.

(3) Click the **OK** button.

Note
The SDK Manager does not currently support SOCKS. The password is protected by the proxy server.

Figure 11 SDK Manager - Configuration Page



Please specify a location for your Download Cache.
New packages will be downloaded here before being installed or added to an offline install.

Download Cache: C:\ProgramData\SCE\SDK Installer\Download Cache

Download Cache Size:2 GB

Delete all files in the download cache.

☐ Use a proxy server for your LAN.

Address: Port:

☑ Enable update notifications

☑ Send usage data to SCE

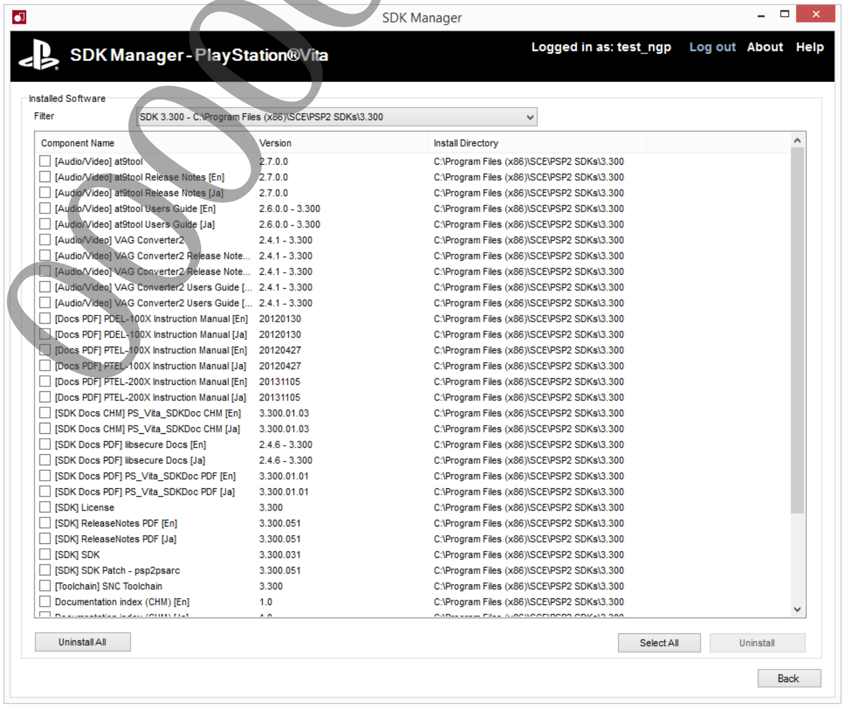
Uninstall SDK

To uninstall components, use **Uninstall SDK** in the SDK Manager home page.

Select a component to be uninstalled from the component list and click **Uninstall button**.

Note
When **Uninstall All** is selected, note that there is a possibility of cross-platform components (such as Visual Studio Integration) being uninstalled.

Figure 12 SDK Manger - Uninstall SDK



Filter: SDK 3.300 - C:\Program Files (x86)\SCE\PSP2 SDKs\3.300

Component Name	Version	Install Directory
☐ [Audio/Video] at9tool	2.7.0.0	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [Audio/Video] at9tool Release Notes [En]	2.7.0.0	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [Audio/Video] at9tool Release Notes [Ja]	2.7.0.0	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [Audio/Video] at9tool Users Guide [En]	2.6.0.0 - 3.300	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [Audio/Video] at9tool Users Guide [Ja]	2.6.0.0 - 3.300	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [Audio/Video] VAG Converter2	2.4.1 - 3.300	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [Audio/Video] VAG Converter2 Release Note...	2.4.1 - 3.300	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [Audio/Video] VAG Converter2 Release Note...	2.4.1 - 3.300	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [Audio/Video] VAG Converter2 Users Guide [...	2.4.1 - 3.300	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [Audio/Video] VAG Converter2 Users Guide [...	2.4.1 - 3.300	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [Docs PDF] PDEL-100X Instruction Manual [En]	20120130	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [Docs PDF] PDEL-100X Instruction Manual [Ja]	20120130	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [Docs PDF] PTEL-100X Instruction Manual [En]	20120427	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [Docs PDF] PTEL-100X Instruction Manual [Ja]	20120427	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [Docs PDF] PTEL-200X Instruction Manual [En]	20131105	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [Docs PDF] PTEL-200X Instruction Manual [Ja]	20131105	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [SDK Docs CHM] PS_Vita_SDKDoc CHM [En]	3.300.01.03	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [SDK Docs CHM] PS_Vita_SDKDoc CHM [Ja]	3.300.01.03	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [SDK Docs PDF] libsecure Docs [En]	2.4.6 - 3.300	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [SDK Docs PDF] libsecure Docs [Ja]	2.4.6 - 3.300	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [SDK Docs PDF] PS_Vita_SDKDoc PDF [En]	3.300.01.01	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [SDK Docs PDF] PS_Vita_SDKDoc PDF [Ja]	3.300.01.01	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [SDK] License	3.300	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [SDK] ReleaseNotes PDF [En]	3.300.051	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [SDK] ReleaseNotes PDF [Ja]	3.300.051	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [SDK] SDK	3.300.031	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [SDK] SDK Patch - psp2psarc	3.300.051	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ [Toolchain] SNC Toolchain	3.300	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300
☐ Documentation index (CHM) [En]	1.0	C:\Program Files (x86)\SCE\PSP2 SDKs\3.300

Switch SDK or Copy Samples

Click **Switch SDK or Copy Samples** on the SDK Manager homepage to adjust setup contents.

Samples and installed SDKs reference directories depending on environment variables. These settings can be easily switched using the SDK Manager.

Copy Sample

To copy the sample directory and switch the `%SCE_PSP2_SAMPLE_DIR%` environment variable, use **Copy Samples**.

SDK Version Switching

To switch the `%SCE_PSP2_SDK_DIR%` environment variable specifying the SDK, use **SDK Switching**.

A list of installed SDKs is shown in the **Installed SDKs** combo box.

You can select another SDK by using this combo box and clicking the **Switch To SDK** button.

Once this operation is confirmed, the SDK Manager will setup the SDK, and the environment variables, and Start menu shortcut to use that SDK. (This may take some time.).

Note
When using the changed environment variables, it is necessary to restart the running application.

Types of Install Directory

Three types of root directories are provided for setting up SDKs.

SCE root directory	Default directory: `%ProgramFiles%\SCE` or `%ProgramFiles(x86)%\SCE`
	This is the root directory to which various tools are installed. If the `%SDK_ROOT_DIR%` environment variable has not been set, it can be specified when the SDK Manager is installed (at the first start-up). Since this directory is set as the `%SDK_ROOT_DIR%` environment variable and it is fixed to allow the operation of tools, do not change this directory. If changing this directory, it is necessary to do the setups of all the tools again. Further, this directory is not subject to SDK version switching.
SDK install directory (SDK directory)	Recommended directory: `%ProgramFiles%\SCE\PSP2 SDKs\<VERSION>` or `%ProgramFiles(x86)%\SCE\PSP2 SDKs\<VERSION>`
	This is the root directory to which the SDK is installed. Normally, the same directory as the SCE root directory is used, but based on a large amount of feedback, selection of an arbitrary directory was made possible. You can specify an arbitrary directory by selecting SDK.
Custom directory	Recommended directory: Displayed in GUI of SDK Manager
	Installation to an arbitrary root directory by specifying a custom directory is possible. To perform installation to a directory other than the "arbitrary directory" in which the package is displayed, click **Customize** and specify a custom install directory.

Verifying the Setup

After a successful installation, by default, the SDK directory is set up with the install directory at the top level. The SCE root directory is set up to `%ProgramFiles%\SCE` by default. The tools are installed to `%SCE_ROOT_DIR%` by default. The following directories may change depending on the status of installed components and the SDK version.

SCE Root Directory			
\PSP2			This platform-related files, the package independent of SDK version.
	+ \I-O Devices		Package related to Input/Output device
	+ \Publishing		Master-related guidelines, etc. Various guidelines including the brand guideline, logo data, NC, TRC, etc., will be installed.
	+ \Tools		Various host tools Applications including the debugger, Neighborhood, Razor, sample browser, Target Manager Server, Publishing Tools, etc., will be installed.
	+ \Services		Windows service program
		+ \Content Manager Assistant for PlayStation(R)Vita DevKit	Application sharing specific folders on the PC with the DevKit/TestKit
		+ \SCE Network Manager for PlayStation(R)Vita DevKit	Program for communication with the DevKit
		+ \USB driver for PSP2 DevKit	USB driver of the DevKit
	+ \System Update Files		System update file
\common			Contains files and applications that can be used in common across platforms
	+ \External Tools		Package dependent on external application
	+ \PSN		PSN℠ related file
	+ \SceVSI		Visual Studio add-in
	+ \File System Driver		Driver for mounting the DevKit's file system

CUSTOM Directory (Modifiable during Installation)		
\Engines		
	+ \PhyreEngine	PhyreEngine™ game engine

SDK component installation to a user defined directory is possible, and multiple versions of SDKs can be installed to arbitrary directories.

The root directory of the active SDK can be obtained with the `SCE_PSP2_SDK_DIR` environment variable, and the structure of the SDK directory is as follows (may vary slightly depending on the SDK version).

SDK Directory (Modifiable during Installation)	
+ \<VERSION>	PlayStation®Vita SDK version directory
+ \host_tools	Applications that run on the development host computer
+ \target	Programs that run on the DevKit
+ \include	Include files
+ \include_common	Include files for multi platforms
+ \src	Source files
+ \sce_data	Data, such as, the dictionary file
+ \sce_module	PRX formatted runtime library
+ \module	Sample programs of PRX formatted runtime library
+ \lib	Libraries
+ \samples	SDK Sample programs
+ \data	Data referenced by multiple samples
+ \sample_code	Sample codes
+ \documentation	Documents
+ **	Language
+ \chm	CHM document
+ \pdf	PDF documents
+ \Hardware_doc	Hardware documents
+ \release_notes	Release Notes (SDK)
+ \SDK_doc	SDK documents
+ \WebAPI_Doc	WebAPI documents
+ \Microsoft_Help_System	Microsoft Help System document source
+ \license	License files
+ \release_notes	Release Notes

Start Menu Shortcuts

Installation using SDK Manager creates shortcuts in the Start menu.

They will mainly be installed in the following types of folders.

Folder	Shortcut File
SCE - Common	Shortcuts of applications and documents common to platforms provided by SCE will be registered
SCE - PS Vita	Shortcuts of applications and documents unique to the PlayStation®Vita platform will be registered

Environment Variables

Check that the environment variables have been correctly installed as follows:

(1) At a Command Prompt window, type **set SCE** and then press **ENTER**.

(2) The environment variables that start with SCE will be displayed from among all of the ones that have been set. Of these, ensure the following environment variables are set.
Note the installation directory may be different:

```
>set SCE <Enter>
SCE_PSP2_FSD=V:
SCE_PSP2_SAMPLE_DIR=C:/Program Files/SCE/PSP2 SDKs/0.900/target/samples
SCE_PSP2_SDK_DIR=C:\Program Files\SCE\PSP2 SDKs\0.900
SCE_ROOT_DIR=C:\Program Files\SCE
```

Manually Setting Environment Variables

The SDK Manager sets the environment variables during installation, but these can be set manually. Note that some developer tools are case sensitive when referring to environment variables.

To manually set environment variables:

(1) Click **Start**, right-click **Computer**, and then click **Properties**.

(2) In the **System Properties** dialog box, on the **Advanced** tab, click **Environment Variables**.

(3) In the **Environment Variables** dialog box, in the **System Variables** section, use the **New** button to configure the following variables, and then click **OK**.

Environment Variable	Value(default)	Description
`SCE_ROOT_DIR`	`C:\Program Files\SCE`	Various tools are placed under the root directory of SCE. This value is set at the installation of SDK Manager.
`SCE_PSP2_SDK_DIR`	`C:\Program Files\SCE\PSP2 SDKs\<VERSION>`	This is the directory of the version number of SDK. This value is set at the installation of SDK.
`SCE_PSP2_SAMPLE_DIR`	`C:/Program Files/SCE/PSP2 SDKs/<VERSION>/target/samples`	This is the sample directory. This value is set at the installation of SDK.

Note
Never change the `%SCE_ROOT_DIR%` **environment variable.** This must be kept constant in order that the various host tools work correctly.

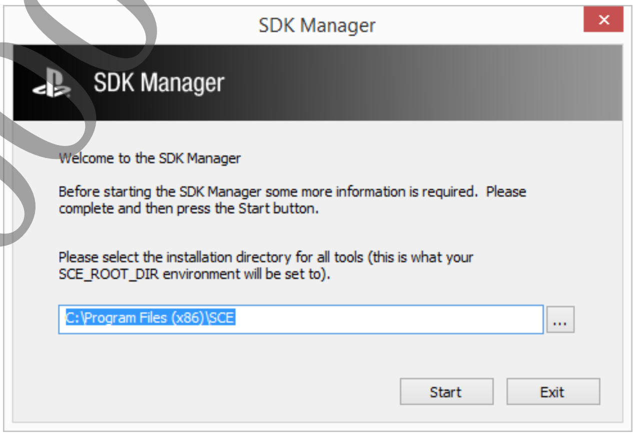
Moreover, SDK Manager sets and manages these environment variables as system environment variables. Therefore, do not over-ride these system environment variables using user environment variables.

When the Environment Variable `%SCE_ROOT_DIR%` Is Deleted

At start-up, the SDK Manager checks the environment variable `%SCE_ROOT_DIR%`. If it has not been set, the dialog for setting the SCE root directory will be displayed. If this dialog is displayed, specify the path and click the **Start** button. Also, even if the SDK Manager is terminated with **Exit** button, you can set the environment variable manually and this dialog displaying at the SDK Manager start-up will stop.

The default installation directory is `%ProgramFiles%\SCE` or `%ProgramFiles(x86)%\SCE`.

Figure 13 Setting of Environment Variable `%SCE_ROOT_DIR%`



Welcome to the SDK Manager

Before starting the SDK Manager some more information is required. Please complete and then press the Start button.

Please select the installation directory for all tools (this is what your SCE_ROOT_DIR environment will be set to).

C:\Program Files (x86)\SCE

3 Connecting, System Update, and Checking Operation

Power on and off, file transfer, and execution can be performed by using Neighborhood for PlayStation®Vita (Neighborhood) on the development host computer connected to the DevKit with a USB cable.

Below is an explanation of the connection method and the operation check method. For details on the operation method, refer to the following "Neighborhood and Utilities User's Guide" document.

```
%SCE_ROOT_DIR%\PSP2\Tools\Neighborhood\help\Neighborhood_and_Utilities-Users_G
uide_e.pdf
```

To control the DevKit with the command line utility, refer to the "Controlling Development Kits by using Utilities" chapter in the "Neighborhood and Utilities User's Guide" document.

Connection Method (DevKit)

The DevKit can be controlled from Windows Explorer when Neighborhood is installed on the development host computer.

To start up Neighborhood, double-click the Neighborhood for PlayStation®Vita icon on the desktop.

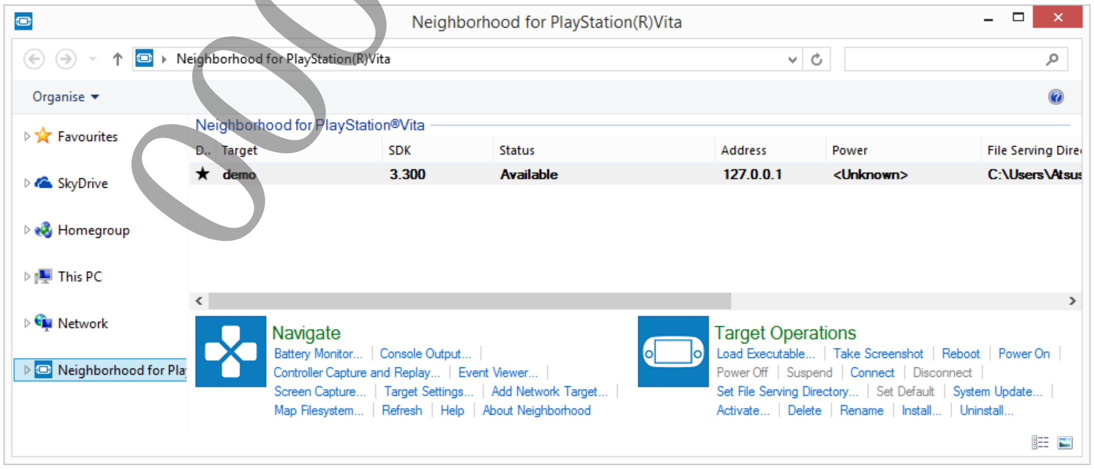
Connect the AC adapter to the DevKit, and connect the USB port (mini-B) for development of the DevKit to the development host computer by using the supplied USB cable. The DevKit has a USB port (mini-B) for development on its lower part. In addition, up to 8 DevKits can be controlled by one development host computer.

Note
By arranging the connection between DevKit and the development host computer, data transfer efficiency between the two can be maintained. Refer to the "Transfer-efficient Connection (DevKit)" section.

If the USB driver has not been installed, install the driver by referring to the "Appendix C: Method to Install Manually the USB Driver" chapter. You may be requested to restart the computer after the USB driver has been installed.

After the installation of the USB driver is completed, the DevKit is added to Neighborhood in Windows Explorer.

Figure 14 Add DevKit to Neighborhood


Neighborhood for PlayStation(R)Vita
Neighborhood for PlayStation(R)Vita
Organise
Favourites
SkyDrive
Homegroup
This PC
Network
Neighborhood for Pla
Navigate
Battery Monitor... | Console Output... |
Controller Capture and Replay... | Event Viewer... |
Screen Capture... | Target Settings... | Add Network Target... |
Map Filesystem... | Refresh | Help | About Neighborhood
Target Operations
Load Executable... | Take Screenshot | Reboot | Power On |
Power Off | Suspend | Connect | Disconnect |
Set File Serving Directory... | Set Default | System Update... |
Activate... | Delete | Rename | Install... | Uninstall...

Neighborhood for PlayStation®Vita

D..	Target	SDK	Status	Address	Power	File Serving Dire
★	demo	3.300	Available	127.0.0.1	<Unknown>	C:\Users\Atsu

Selecting the DevKit

Open Neighborhood, select the displayed DevKit and select **Default settings** in the **Target Operations** category. If the DevKit is not displayed, refer to the "Appendix C: Method to Install Manually the USB Driver" chapter, and perform updating the USB driver.

If using a DevKit connected to a remote development host computer, choose **Add Network Target** in the **Navigate** category, and input the IP address or the host name of the development host computer to which the DevKit is connected.

Connecting DevKit

Select the DevKit listed in Neighborhood and select **Connect** in the **Target Operations** category.

If the connection is established, connection status is changed from <Available> to **<USERNAME@HOST-NAME, TargetManager>** (USERNAME and HOST-NAME vary according to the usage environment). Under normal use, the DevKit will be in the connected status automatically, but if the connection was canceled, reconnection can be done with the above-described method.

If the DevKit is being used by another user, this can be determined by the USERNAME and HOST-NAME.

System Update

The flash memory of the DevKit contains the kernel, system software and the Communication Processor (CP) controlling communication with the development host computer. These components may need to be updated. It is possible for the DevKit to update them simultaneously with a single system update file.

For the system update, the following two methods are mainly used.

Advantages of each method are explained below.

Note
A system update file containing a specific DevKit's model number means that the file is released only for the DevKit. Therefore, such system update file cannot be used for another DevKit.

System update using the development host computer (DevKit)

This method is effective when the DevKit is connected to the development host computer via USB. PUP file downloaded from the PlayStation®Vita DevNet can be used by transferring the file via USB. For setting up the SDK, use this method as a rule. With this method, Neighborhood can be used.

System update using the settings application of the system software (DevKit/TestKit)

In this method, a system update is performed on the system software. This is effective when connection cannot be established between the development host computer and the DevKit or the TestKit. In addition, by changing the settings, it becomes possible to refer to system update file (PUP) on the web server in the local area network and to perform a system update.

This method includes a method with which a system update file is referenced via network by connecting to a PC or PlayStation®3.

For details, refer to the "Appendix B: Other System Update Methods" chapter.

Checking the Version (DevKit/TestKit)

If you check the version before performing a system update, follow the procedure on the system software GUI shown below.

- **Settings** > **System** > **System Information**

Updating DevKit with Neighborhood (DevKit)

Updating with Neighborhood is done as follows.

(1) Open Neighborhood and select the DevKit

(2) Select **System Update** in the **Target Operations** category.
Select the update file installed at the location below and click **OK**.

```
%SCE_ROOT_DIR%\PSP2\System Update Files\PSP2UPDAT-<VERSION>(-<MODEL
NAME>).PUP
```

Alternatively, drag and drop the above system update file on the DevKit listed in Neighborhood.

(3) Power on the DevKit.
As the DevKit is powered off after the update, power it on again by clicking **Power On** in the **Target Operations** category.

(4) After the DevKit starts up, check the version as described in "Checking the Version (DevKit/TestKit)"

System Update by Command Line of the Development Host Computer (DevKit)

Follow the procedure described below when performing a system update of the DevKit by the command line. The system update file is installed in the following location:

```
"%SCE_ROOT_DIR%\PSP2\System Update Files\PSP2UPDAT-<VERSION>(-<MODEL
Name>).PUP"
```

(1) Run psp2ctrl.

1-1. Open a Command Prompt window.

1-2. At the command prompt, type the following command, and then press ENTER.
The update is started by executing this command.

```
>psp2ctrl update "%SCE_ROOT_DIR%\PSP2\System Update
Files\PSP2UPDAT-<VERSION>(-<MODEL Name>).PUP"
```

(2) Power on the DevKit.
As the DevKit is powered off after the system update, power it on again.
At the command prompt, type **psp2ctrl on** and then press ENTER.

```
>psp2ctrl on
```

(3) After the DevKit starts up, check the version as described in "Checking the Version (DevKit/TestKit)".

Operation Check of the Development Environment (Building and Executing Sample Programs) (DevKit)

This section explains the procedures to build a sample program and execute it on the DevKit. If the sample program operates normally, this indicates that the system has been set up correctly.

Check the sample browser for details on each sample.

- **Start > All Programs > SCE - PS Vita > Sample Browser for PS Vita**

Build and Execute a Sample Program

The basic sample displays the text "`Hello, PSP2 World!`" on the console output.

To build and execute the basic sample:

(1) Open Visual Studio.

(2) In Visual Studio, open the following solution:

```
"%SCE_PSP2_SDK_DIR%\target\samples\sample_code\system\tutorial_startup\
basic\basic.sln"
```

(3) Build on Visual Studio.
Ensure Visual Studio Integration has been installed, and then on the **Build** menu, click **Build Basic**. After a successful build, the following executable file is created (*** depends on the solution configuration of Visual Studio):

```
"%SCE_PSP2_SDK_DIR%\target\samples\sample_code\system\tutorial_startup\
basic\***\basic.self"
```

(4) Execute the sample by the following methods:
On the **Debug** menu of Visual Studio, click **Start Without Debugging**. The SELF file will be transferred to the connected DevKit and executed.
Verify that the following character strings are displayed:

```
Hello, PSP2 World!
## tutorial_startup/basic: FINISHED ##
```

Executing Using Neighborhood (DevKit)

To execute a program using Neighborhood:

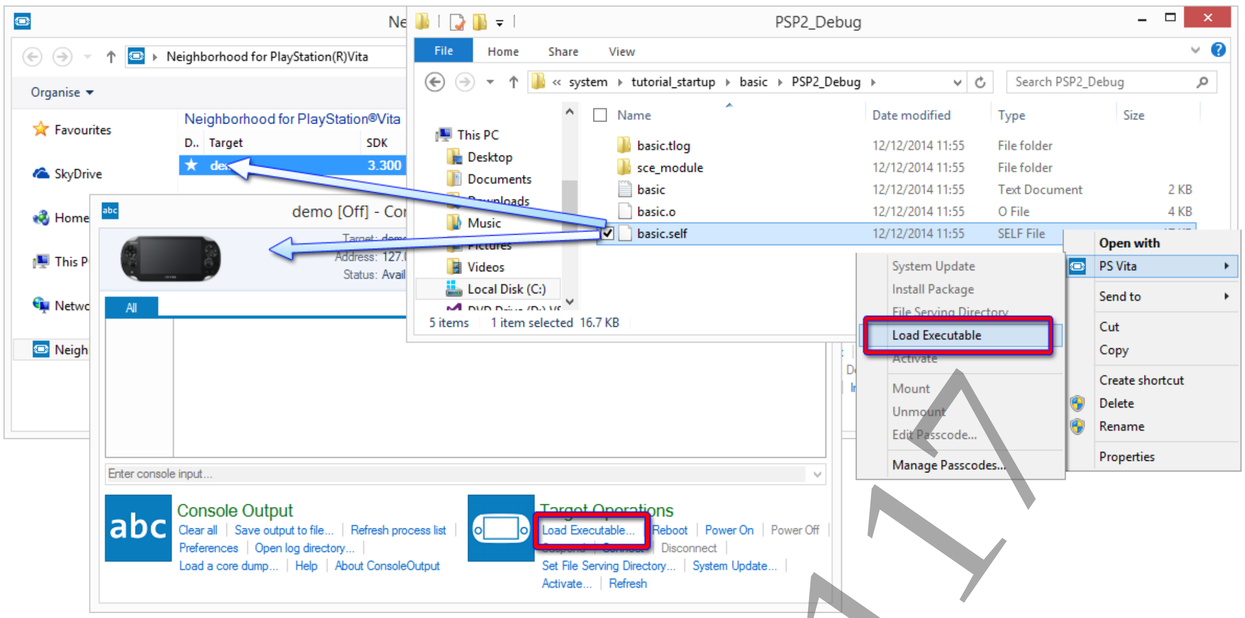
(1) Select the DevKit with Neighborhood.

(2) Open the console output window by clicking **Console Output** in the **Navigate** category.

(3) Click **Load Executable** in the **Target Operations** category, and select the SELF file to be executed.
In addition, the execution is possible from Neighborhood with the following operations:

- Drag and drop the SELF file on the **Console Output** window
- Drag and drop the SELF file on the target list of Neighborhood.
- Right-click the SELF file, and select **Default PS Vita > Load Executable** from the displayed context menu.

Figure 15 Various Execution Method in Neighborhood



Other Execution Methods

The following methods are available for the execution of a program.

Execute from Visual Studio

- Press CTRL + F5 (Shortcut for **Debug** > **Start Without Debugging**)

Sample Browser

- Click **Run** on the sample browser.

Execute a Sample Program on the Command Line

To execute the sample program on the command line:

(1) Connect to the DevKit.
   1-1. Open a Command Prompt window.
   1-2. At the command prompt, type **psp2ctrl connect** and then press **ENTER**.

```
> psp2ctrl connect
```

(2) Power on the DevKit.
At the command prompt, type **psp2ctrl on** and then press **ENTER**.

```
> psp2ctrl on
```

(3) Execute the sample program file.
At the command prompt, type the following command, and then press **ENTER**:

```
> psp2run /reboot /console /elf "***.self"
```

Configuring the DevKit Using Neighborhood (DevKit)

(1) Start Neighborhood.
Start "Neighborhood for PlayStation®Vita" on the desktop and select the DevKit whose settings you wish to edit.

(2) Click **Target Settings** of the **Navigate** category.

> **Note**
> Do not power off the DevKit during settings configuration with Neighborhood.

(3) Click **Refresh** in the **Settings Operation** category of the **Settings** dialog to load the settings. This can take a while.

For details on the settings, refer to the "Development Kit Neighborhood Settings Guide" document.

Set the Name of the DevKit

For the settings to be edited here, we recommend setting the DevKit's name.

Turn off the Power Supply with Neighborhood (DevKit)

To power off the DevKit

(1) Open Neighborhood, select the **default PSP2**, and click **Power Off** in the **Target Operations** category.

(2) Unplug the AC adaptor from the DevKit system.

(3) Pull out the USB cable used for connecting the DevKit and the development host computer.

Turn off the Power Supply on the System Software (DevKit/TestKit)

To power off the DevKit and the Test Kit

(1) Press the power button of the DevKit or the TestKit for 2 seconds.

(2) Tap **Power Off**.

Updating the Expiry Time Using Neighborhood (DevKit)

When the expiry time of the DevKit has elapsed, the expiry time must be updated by applying the activation file (The expiry time is displayed in the Neighborhood property).

Apply the activation file (.afv) by dragging and dropping the file on the Neighborhood target.

Updating the Expiry Time of DevKit and TestKit Using CMA (Activate by Connecting PC) (DevKit/TestKit)

With Content Manager Assistant for PlayStation®Vita DevKit (CMA), activation can be executed, referring to the activation file stored on the PC side.

Place "`vita_activation_XXXXXXXXXXXX.afv`", which is an activation file obtained from DevNet, under the following directory of a PC on which CMA has already been installed.

"**My Documents\PS Vita**`\package\activate\vita_activation.afv`"

("**My Documents\PS Vita**" is a path defined with **Application/Backup File** of CMA settings.)

Connect the PC to the multi-use port of DevKit or TestKit (PTEL-1000 series) or to the USB port of TestKit (PTEL-2000 series) using a USB cable.

In the case of the TestKit, the time must be set via the internet before activation execution. Set the time by selecting **Date & Time** > **Date & Time Settings** > **Set via Internet** > **Set Now** in the Settings application of the system software.

Select **Settings** > ★**Debug Settings** > **Activation > Activate by Connecting PC** of the DevKit or TestKit to execute activation using the above file. If the activation is successfully done, the DevKit or TestKit will restart and then it can be checked that the expiry time is updated by confirming **Settings** > ★**Debug Settings** > **Activation** > **Show Expiration Date**.

In addition to the above method, activation can be executed by copying the activation file to the memory card. For details on activation, refer to the "DevKit/TestKit Activation User's Guide" document.

How to Use Memory Cards

In order to be used, the memory card must be formatted in advance.

(To use the memory card, update the system software version to version 0.990 or above)

Use the File System Driver or the psp2ctrl command line utility to access formatted memory cards.

Formatting Memory Cards (DevKit/TestKit)

Follow the procedure below to format the memory card from the system software GUI.

- **Settings** > **Format** > **Format Memory Card**

By formatting, a data directory will be created on the memory card, and it will be possible to use it for the following purposes:

- Saving applications for start-up from ★APP_HOME
- Accessing save data (For details on the save data, refer to the "Save Data User's Guide" document.)
- Outputting core files (For details on the core file, refer to the "Core Dump Overview" document.)
- Activation (For details on the activation, refer to the "DevKit/TestKit Activation User's Guide" document.)
- Various types of file access

If data access is attempted without formatting, the following error message will be output to the command prompt.

```
[ERROR]: RDRFP_ERROR_ENOENT 0x12002
```

Copying Files Using the psp2ctrl Command Line Utility (DevKit)

Follow the procedure below when using the psp2ctrl command line utility.

```
> psp2ctrl cp C:\temp\file.dat ux0:data
```

In addition, it is possible to perform other operations such as directory creation (psp2ctrl mkdir) and deletion (psp2ctrl rm) on the memory card.

By using the series of commands in the example below to copy files, it is possible to reference update data during updates.

```
> psp2ctrl mkdir ux0:data/PSP2
> psp2ctrl mkdir ux0:data/PSP2/UPDATE
> psp2ctrl cp "C:\Program files\SCE\PSP2\System Update
Files\PSP2UPDAT-0_995_000.PUP"
ux0:data/PSP2/UPDATE/PSP2UPDAT.PUP
[INFO]: 'C:\Program files\SCE\PSP2\System Update
Files\PSP2UPDAT-0_995_000.PUP' ->
'ux0:data/PSP2/UPDATE/PSP2UPDAT.PUP'
```

For details on the `psp2ctrl` command line utility, refer to the "Neighborhood and Utilities User's Guide" document for Neighborhood.

File System Mapping (DevKit)

In development host computers where the File System Driver has been installed, it is possible to assign the DevKit as a drive on the development host computer. For the assigned drive, file management can be performed in the same manner as for normal file systems from Windows Explorer.

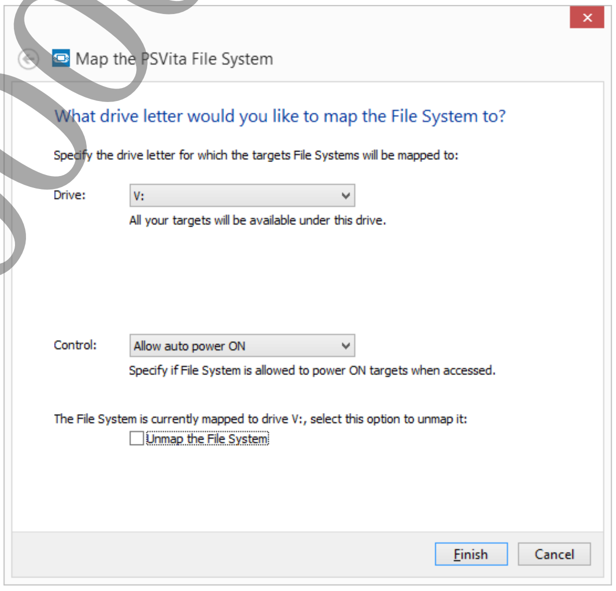
Such "file system mapping" can be performed from Neighborhood and the psp2ctrl command line utility.

When mapping the PlayStation®Vita file system, check that the name of the DevKit system has been set. The system name will be displayed immediately below the mapped drive. It will not be displayed if it has not been set. You can set the system name using the settings tool of Neighborhood.

File System Mapping with Neighborhood

Connect the DevKit, and, in powered-on state, click **Map Filesystem** in the **Navigate** category to assign the drive letter with the file system map dialog.

Figure 16 PlayStation®Vita File System Mapping



Map the PSVita File System

What drive letter would you like to map the File System to?

Specify the drive letter for which the targets File Systems will be mapped to:

Drive: V:

All your targets will be available under this drive.

Control: Allow auto power ON

Specify if File System is allowed to power ON targets when accessed.

The File System is currently mapped to drive V:, select this option to unmap it:

☐ Unmap the File System

Copying Files

To copy files between the development host computer and the DevKit, use PlayStation(DRIVE:)\<System Name> displayed on Windows Explorer.

Mapping the File System with the psp2ctrl Command Line Utility

When using the psp2ctrl command line utility, assign a drive letter by following the procedure below:

```
> psp2ctrl map <DRIVE>
```

To assign the DevKit to drive E, specify as follows:

```
> psp2ctrl map E
```

For copying methods using the psp2ctrl command line utility, refer to the preceding section, "Copying Files Using the psp2ctrl Command Line Utility (DevKit)".

Transfer-efficient Connection (DevKit)

By arranging the connection between DevKit and the development host computer, the time entailed in the transfer of an in-development program to DevKit can be shortened.

DevKit and the development host computer are connected using a USB bus. Because the USB uses a bus-structure communication technology, limited communication bandwidth is shared by all the USB devices that are connected at the same time. Therefore, an increase in the number of connected USB devices leads to a decrease in each device’s transfer speed.

A USB device that may appear to not be in communication may actually be communicating. For example, a keyboard or a mouse communicates with the development host computer every 1 to 8 ms even if it is untouched. DevKit also communicates with the development host computer for state checks.

Thus, the less USB devices that share the USB bus with DevKit, the more DevKit can use the communication bandwidth and accelerate communication with the development host computer. Data transfer is fastest when one DevKit is connected to one USB bus.

Checking for Connected USB Devices

Because it is difficult to visually check for USB devices that are sharing one USB bus, the Windows Device Manager can be used to perform this check.

The method to check for USB devices sharing one USB bus is as follows.

(1) Start up Device Manager (devmgmt.msc). The Device Manager can be started from the **Control Panel** or the **Manager** menu displayed by right-clicking on the **Computer** icon.

(2) Select the **View > Devices by connection**.

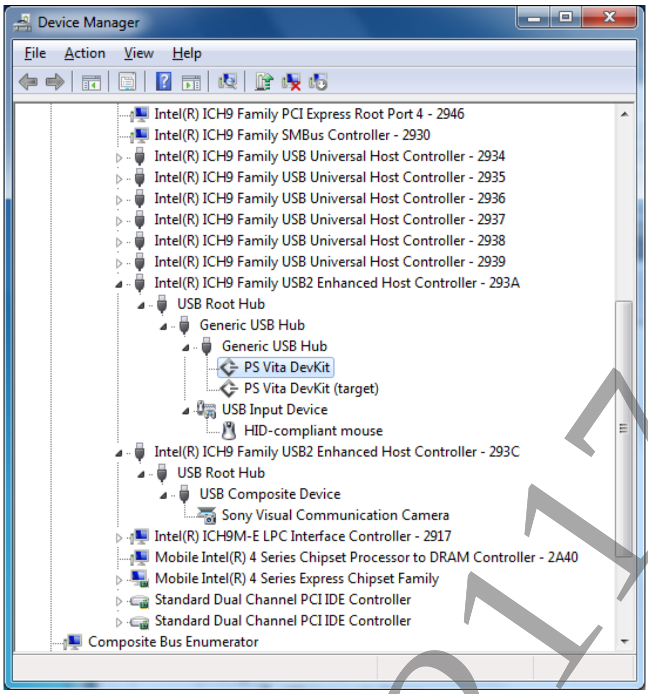
(3) Select and click the icon with the computer name.

(4) Press the asterisk key of the keypad (ten key) section on the keyboard to display all devices.

(5) Search for devices with *** **Enhanced Host Controller** (the *** section represents, for example, "PCI to USB"). Each item with this display comprises one USB bus. **Open Host Controller** and **Universal Host Controller** each comprises one USB bus; however, they are not appropriate for DevKit connection. DevKit automatically avoids them when making a connection.

(6) Check the USB devices connected to a USB bus and adjust positions of the USB port to which USB devices and DevKit are connected so that other USB devices will not be connected to the USB bus to which DevKit is connected.

Figure 17 Device Manager with DevKit Connected



```
Intel(R) ICH9 Family PCI Express Root Port 4 - 2946
Intel(R) ICH9 Family SMBus Controller - 2930
Intel(R) ICH9 Family USB Universal Host Controller - 2934
Intel(R) ICH9 Family USB Universal Host Controller - 2935
Intel(R) ICH9 Family USB Universal Host Controller - 2936
Intel(R) ICH9 Family USB Universal Host Controller - 2937
Intel(R) ICH9 Family USB Universal Host Controller - 2938
Intel(R) ICH9 Family USB Universal Host Controller - 2939
Intel(R) ICH9 Family USB2 Enhanced Host Controller - 293A
  USB Root Hub
    Generic USB Hub
      Generic USB Hub
        PS Vita DevKit
        PS Vita DevKit (target)
      USB Input Device
        HID-compliant mouse
Intel(R) ICH9 Family USB2 Enhanced Host Controller - 293C
  USB Root Hub
    USB Composite Device
      Sony Visual Communication Camera
Intel(R) ICH9M-E LPC Interface Controller - 2917
Mobile Intel(R) 4 Series Chipset Processor to DRAM Controller - 2A40
Mobile Intel(R) 4 Series Express Chipset Family
Standard Dual Channel PCI IDE Controller
Standard Dual Channel PCI IDE Controller
Composite Bus Enumerator
```

Figure 17 shows a display example of the Device Manager. There are two USB buses (***USB2 Enhanced Host Controller**) and a DevKit (**PS Vita DevKit**) and mouse (**HID-compliant mouse**) are connected to the same USB bus. In this case, DevKit can dominate one USB bus and its data transfer speed can be accelerated by moving the mouse to the other USB bus.

Increasing USB Buses

When only one USB bus is available on the development host computer, an additional USB installing card can be connected to the development host computer and one DevKit can be connected to it. This method is also effective with development host computers that already have multiple USB buses.

Reference Information: USB Device Communication Load

Some USB devices entail more communication load than others. When you cannot prevent the sharing of a USB bus, make sure the DevKit is shares the USB bus with a USB device that entails a light communication load in order to maintain the data transfer efficiency of DevKit. Communication load for each USB device is as follows.

DevKit has two kinds of USB connection: one that uses the USB port for debugging and one that uses the multi-use port. Each is noted, respectively, as DevKit (USB port) and DevKit (multi-use port).

USB Devices with Very Heavy Communication Load

- USB Ethernet, USB Wireless LAN, USB Bluetooth®
- USB camera (Web camera)
- USB audio (USB speakers, etc.)
- DevKit (USB port)
- Data-transferring USB memory, USB HDD, USB card reader
- Data-transferring DevKit (multi-use port), TestKit

USB Devices with Heavy Communication Load

- USB mouse, USB keyboard
- USB hub

USB Devices with Light Communication Load

- USB memory, USB HDD, USB card reader (not transferring data)
- DevKit (multi-use port, TestKit (not transferring data)

USB Devices without Communication

- USB fan
- Mobile phone connected with a battery charging cable

Note DevKit itself is a USB device that entails a very heavy communication load. When connecting multiple DevKits, make sure to connect each DevKit to a separate USB bus.

4 Troubleshooting

This chapter explains how to solve DevKit-related or TestKit-related problems that may occur.

Method to Terminate Target Manager Server for PlayStation®Vita DevKit

Currently there is no method to explicitly terminate Target Manager Server for PlayStation®Vita DevKit.

To terminate Target Manager Server for PlayStation®Vita DevKit, in Task Manager, end the Task Manager Target Manager Server for PlayStation®Vita DevKit (psp2tm) process.

Start Network Manager for PlayStation®Vita DevKit.

Network Manager for PlayStation®Vita DevKit normally starts as a Windows service, but if it does not start:

(1) In Control Panel, open **Administrative Tools**, and then double-click **Services**.

(2) In Services, start **SCE Network Manager for PlayStation®Vita DevKit**.

If an Error Occurs During System Software Update (DevKit/TestKit)

From system software 1.500 onward, multiple system software update files are being offered. If the following error occurs during updating with Neighborhood, check whether the system software update file is correct.

Operation failed (API Error Target error - LOADP command failed but target did not give reason)

Also, a message such as the following is displayed on the system software GUI.

An error has occurred. (C2-12848-3)

System Software Cannot Be Updated Using Normal Procedures (DevKit/TestKit)

When the system software does not startup even though the DevKit or the TestKit is powered on or cannot be updated using normal procedures, it is possible to execute these procedures by starting up the DevKit or the TestKit in safe mode. For the safe mode startup method and processing method, refer to the "Safe Mode Features" chapter in the "System Software Overview" document.

The Expiry Time Has Elapsed (DevKit/TestKit)

DevKits and TestKits have a set expiry time.

When a message prompting for activation is displayed on the bottom left of the home screen, or the expiry time displayed in the Neighborhood property has elapsed, update the expiry time by applying the activation file. For the DevKit, it is possible to simultaneously apply the activation file to multiple targets of Neighborhood by dragging and dropping the file. For the TestKit, the activation file can be applied through CMA. For details, refer to the "DevKit/TestKit Activation User's Guide" document.

Failed Application Start-up with "Invalid working directory specified" Message

When one of the two directories below is specified as a working directory during application runtime using the file path setting file or Neighborhood, a message stating, "`Invalid working directory specified`" will be displayed, and application start-up will fail.

- `ux0:data`
- `ux0:data/savedata`

In this case, execute the application specifying a directory other than the above as a working directory.

When Update or Activation Using CMA Cannot Be Executed

CMA for development will not automatically start. It must be manually started up from the following shortcut.

Start > All Programs > SCE - PS Vita > CMA for PS Vita DevKit

Also, if the version of CMA does not correspond to the version of system software, it is not possible to perform connection or operation using CMA application even when the DevKit or TestKit is connected. To use the CMA application, check the version information by right-clicking the CMA icon on the task tray, and use the version corresponding to that of the system software with reference to the DevNet's Technical Notes ID 201203-07 (https://psvita.scedev.net/technotes/view/267). Do not use Content Manager Assistant for PlayStation®, and use Content Manager Assistant for PlayStation®Vita DevKit released as the SDK instead.

5 Document to Be Read after Completing Setup

After the setup is completed, the following document should be read. There are also a number of related documents and references, to be referred to as needed.

Information Resource Guide:

The "Information Resource Guide" document provides information that is useful to know for application development.

`%SCE_PSP2_SDK_DIR%\documentation\en\pdf\SDK_doc\1st_read\Information_Resource_e.pdf`

Hardware Overview:

The "Hardware Overview" document provides technical specifications as well as practical information useful for application development.

`%SCE_PSP2_SDK_DIR%\documentation\en\pdf\SDK_doc\Development_Environment\Hardware-Overview_e.pdf`

SDK Release Notes:

`%SCE_PSP2_SDK_DIR%\documentation\en\pdf\release_notes\<VERSION>\SDK_ReleaseNotes_<VERSION>.pdf`

DevKit/TestKit Activation User's Guide:

The "DevKit/TestKit Activation User's Guide" document provides an introduction on the method for updating the expiry time of the DevKit and the TestKit and the method for managing them, etc.

`%SCE_PSP2_SDK_DIR%\documentation\en\pdf\SDK_doc\1st_read\DevKit_TestKit_Activation-Users_Guide_e.pdf`

Links to PlayStation®Vita Developer Network:

Technical Notes

https://psvita.scedev.net/technotes/

Roadmap

https://psvita.scedev.net/docs/roadmap/

PS Vita Tutorial Videos

https://psvita.scedev.net/docs/training_videos/

6 Tutorials for Application Creation/Debugging/Performance Analysis

This chapter uses tutorials to explain application creation, debugging, and performance analysis to beginners to PlayStation®Vita application development.

The Visual Studio screens, section names, operation procedures, etc. shown in this chapter are based on Visual Studio 2013. If using another version, there may be differences.

Preparation

Before executing the tutorials in this chapter, first perform the following procedures.

Ensure Your Default DevKit is Set Up and Available

(1) Connect the USB development port (mini-B) of the DevKit to the development host computer by using the USB cable supplied with the DevKit.

(2) Click on **Neighborhood for PlayStation(R)Vita** in the Windows Explorer Navigation Pane.

The DevKit is automatically detected and set to the default DevKit. For more information on setting up the default DevKit, refer to the "Connecting, System Update, and Checking Operation" chapter.

Enable the Option to Display Console Output in the Output Window

(1) In Visual Studio, select **Tools** > **Options**.

(2) In the **Options** dialog:

- In the left pane, expand **ProDG VSI**, and then click **General**.
- In the right pane, select the **Capture TTY to Output window on Start Without Debugging** check box, and then click **OK**.

Creating and Executing an Application

This tutorial provides tips for PlayStation®Vita development by displaying procedures for creating a simple sample application. This sample outputs the string `"Hello, PSP2 World!"` to the console output.

A complete set of files for this tutorial is available at the following location:

```
%SCE_PSP2_SAMPLE_DIR%\sample_code\system\tutorial_startup\basic
```

An outline of the execution procedure for this tutorial is as follows.

(1) Create a New Project
(2) Add a Source File to the Project
(3) Edit the Source File
(4) Build the Project
(5) Load basic.self onto the DevKit and Execute It

(1) Create a New Project

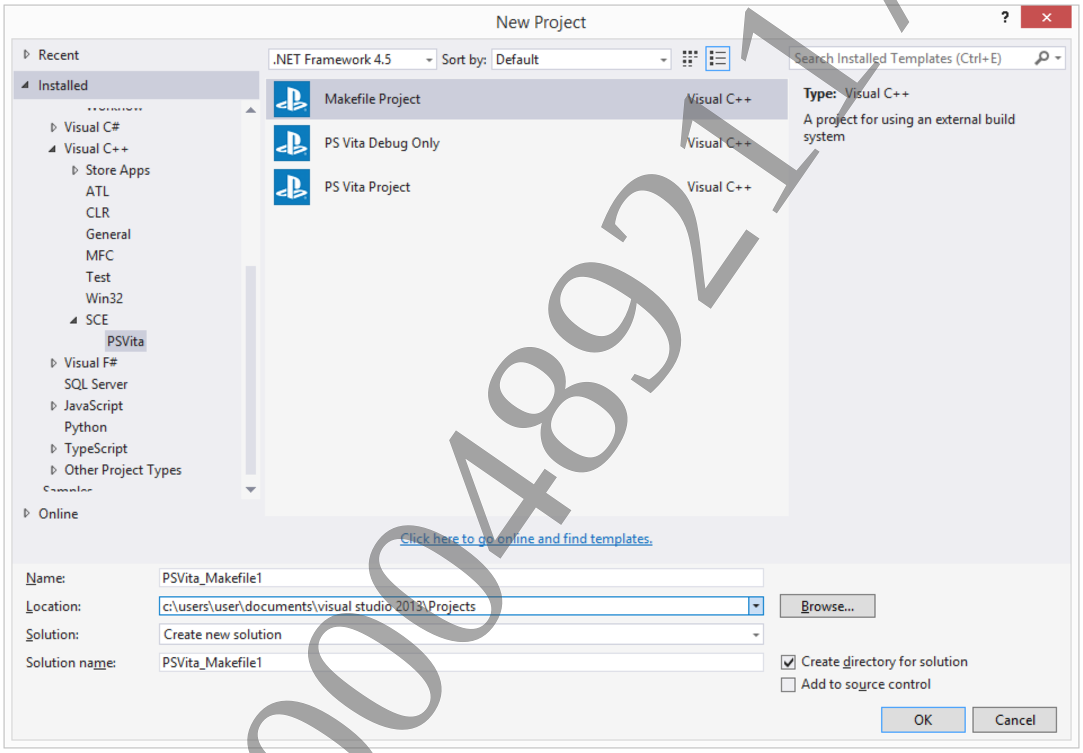
Use the following procedure to create a new PS Vita SELF project called "basic".

(1) Select **File** > **New** > **Project**.

(2) In the **New Project** dialog:

- In the left pane, select **Installed Templates**, select **Visual C++** > **SCE** > **PS Vita**.
- In the middle pane, select the **PS Vita Project** template.
- In the **Name** box, type **basic**.
- Change the location if required in the **Location** box, and then click **OK**.
- Click **Next**, and then clear the **Use Precompiled header** check box.
- Click **Finish**.

Figure 18 New Project Dialog



New Project

Recent
Installed
Visual C#
Visual C++
Store Apps
ATL
CLR
General
MFC
Test
Win32
SCE
PSVita
Visual F#
SQL Server
JavaScript
Python
TypeScript
Other Project Types
Online

.NET Framework 4.5 | Sort by: Default | Search Installed Templates (Ctrl+E)

Makefile Project	Visual C++
PS Vita Debug Only	Visual C++
PS Vita Project	Visual C++

Type: Visual C++
A project for using an external build system

Click here to go online and find templates.

Name: PSVita_Makefile1
Location: c:\users\user\documents\visual studio 2013\Projects
Solution: Create new solution
Solution name: PSVita_Makefile1
Create directory for solution [✓]
Add to source control [ ]

The project is created with two configurations, "Debug" and "Release".

(2) Add a Source File to the Project

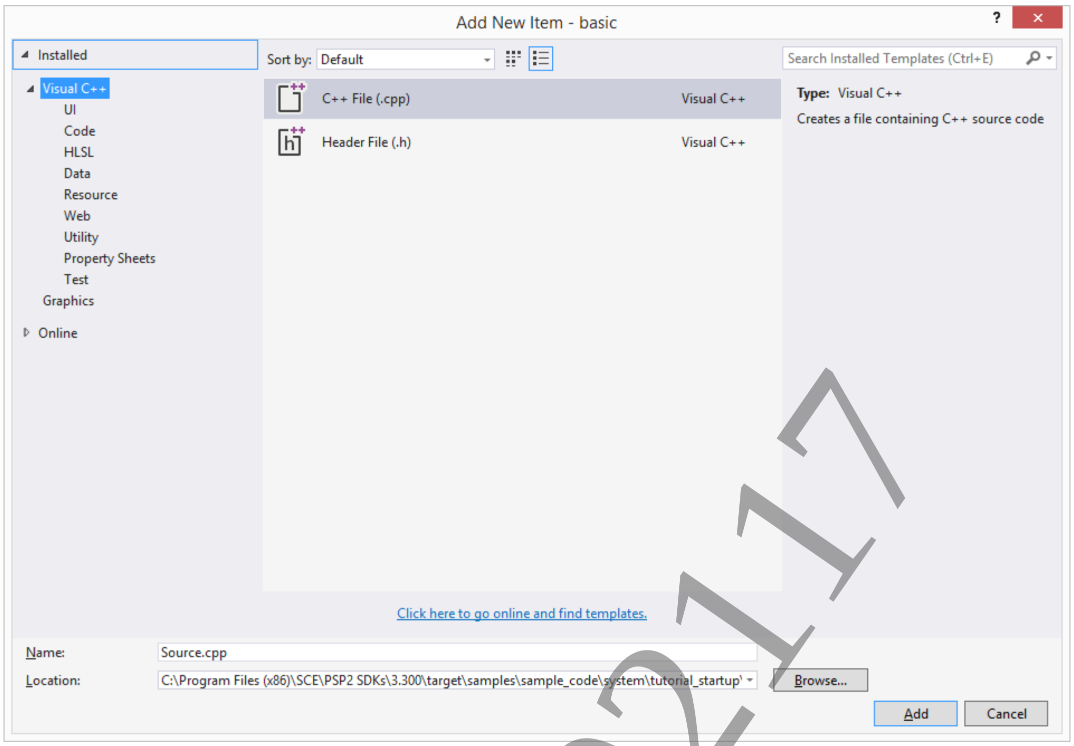
Add the basic.cpp file to the project in Solution Explorer. Use the following procedure.

(1) In **Solution Explorer** pane, right-click the **Source Files** folder, select **Add** > **New Item**.

(2) In the **Add New Item** dialog:

- In the left pane, select **Visual C++**.
- In middle pane, select **C++ File (.cpp)**.
- In the **Name** box, type **basic** and then click **Add**.

Figure 19 Add New Item Dialog


Add New Item - basic
Installed
Visual C++
UI
Code
HLSL
Data
Resource
Web
Utility
Property Sheets
Test
Graphics
Online
Sort by: Default
Search Installed Templates (Ctrl+E)
C++ File (.cpp) Visual C++
Header File (.h) Visual C++
Type: Visual C++
Creates a file containing C++ source code
Click here to go online and find templates.
Name: Source.cpp
Location: C:\Program Files (x86)\SCE\PSP2 SDKs\3.300\target\samples\sample_code\system\tutorial_startup\
Browse...
Add
Cancel

A new C++ file named basic.cpp is added to the project in the Source Files folder.

(3) Edit the Source File

Add a simple `main()` function to the source file that prints text to the console output. Input the following source code in basic.cpp and save the file.

```
#include <stdio.h>

int main(void)
{
    printf("Hello, PSP2 World!\n");
    return 0;
}
```

(4) Build the Project

Select **Build** > **Build Solution**.

Note
When you perform the build, certain run-time libraries that are required to run the executable are implicitly linked. For example, libraries that include startup code, the stub library libSceKernel_stub.a which allows for access to public kernel APIs, and standard C libraries will be linked.

When the build is successful, the executable file basic.self is created.

(5) Load basic.self onto the DevKit and Execute It

Select **Debug** > **Start Without Debugging**. This will load basic.self onto the DevKit, and execution will start immediately.

When basic.self is correctly executed, the Output window will display "`Hello, PSP2 World!`".

Debugging an Application

This tutorial explains the procedure for building a multi-thread application in Visual Studio and performing debugging..

A complete set of files for this tutorial is available at the following location:

```
%SCE_PSP2_SAMPLE_DIR%\sample_code\system\api_libatomic
```

An outline of the execution procedure for this tutorial is as follows.

(1) Open the Sample Solution

(2) Open the main.c File and Confirm the Processing in the `parallelForEach()` Function

(3) Build the Solution

(4) Debug sample_libatomic.self

(1) Open the Sample Solution

Open the following Visual Studio solution.

```
%SCE_PSP2_SAMPLE_DIR%\sample_code\system\api_libatomic\sample_libatomic.sln
```

(2) Open the main.c File and Confirm the Processing in the `parallelForEach()` Function

In **Solution Explorer** pane, expand the **sample_libatomic** project, expand the **Source Files** folder, and then double-click **main.c**. This will display the main.c source code in the main window.

In order to understand the application behavior when performing the debugging in step (4), confirm the following processing in the `parallelForEach()` function in main.c.

- Creates three threads by calling `sceKernelCreateThread()`.
- Starts each thread by calling `sceKernelStartThread()`.
- Waits for each thread to complete by calling `sceKernelWaitThreadEnd()`.
- Deletes each thread by calling `sceKernelDeleteThread()`.

(3) Build the Solution

Select **Build** > **Build Solution**. The executable file sample_libatomic.self is now created.

(4) Debug sample_libatomic.self

Set a breakpoint in the while loop of the `worker_thread_entry()` function and debug sample application (sample_libatomic.self). Perform the following procedure.

(1) Click on a breakpoint marker for one of the lines in the `worker_thread_entry()` function while loop or place the curser in a line and press the F9 key.

(2) Select **Debug** > **Start Debugging** to start debugging the sample.

(3) When debugging starts, execution will stop at the breakpoint. Here, the F11 key will step through the function, the F10 key will step over the function, and the SHIFT+F11 key will step out of the function.

Note
When **Threads** is selected after selecting **Debug** > **Windows**, it will be possible to display thread information. When **Modules** is selected, module information will be displayed. In addition, kernel object information can be displayed by selecting **Kernel Objects**.

(4) When debugging is complete, select **Debug** > **Stop Debugging** to stop debugging.

Debugging a Running Application

By attaching a running process to Visual Studio, it will be possible to debug a running application with Visual Studio. This tutorial explains the procedure. In addition, the procedure for saving a core dump file while an application is running and the procedure for resuming debugging by loading a saved core dump file in Visual Studio are also shown.

Note
A core dump file is a file that includes information (such as the memory content, contexts, and thread states at the time of the dump) useful for analyzing the causes of application/kernel problems. For details on core dump features, refer to the "Core Dump Overview" document.

A complete set of files for this tutorial is available at the following location:

```
%SCE_PSP2_SAMPLE_DIR%\sample_code\system\api_libatomic
```

An outline of the execution procedure for this tutorial is as follows.

(1) Open the Sample Solution File
(2) Correct the Source Code
(3) Build the Solution
(4) Execute sample_libatomic.self Normally
(5) Attach a Running Process and Perform Debugging
(6) Save a Core Dump File
(7) Load the Sample Core Dump File and Perform Debugging

(1) Open the Sample Solution File

Open the following Visual Studio solution file.

```
%SCE_PSP2_SAMPLE_DIR%\sample_code\system\api_libatomic\sample_libatomic.sln
```

(2) Correct the Source Code

Correct the source code so that the executable file does not terminate immediately after execution starts.

Insert the following source code at the start of the `worker_thread_entry()` function while loop.

```
sceKernelDelayThread(10 * 1000 * 1000);
```

This will delay the `worker_thread_entry()` function while loop thread for 10 seconds.

(3) Build the Solution

Select **Build** > **Build Solution**. The executable file sample_libatomic.self is now created.

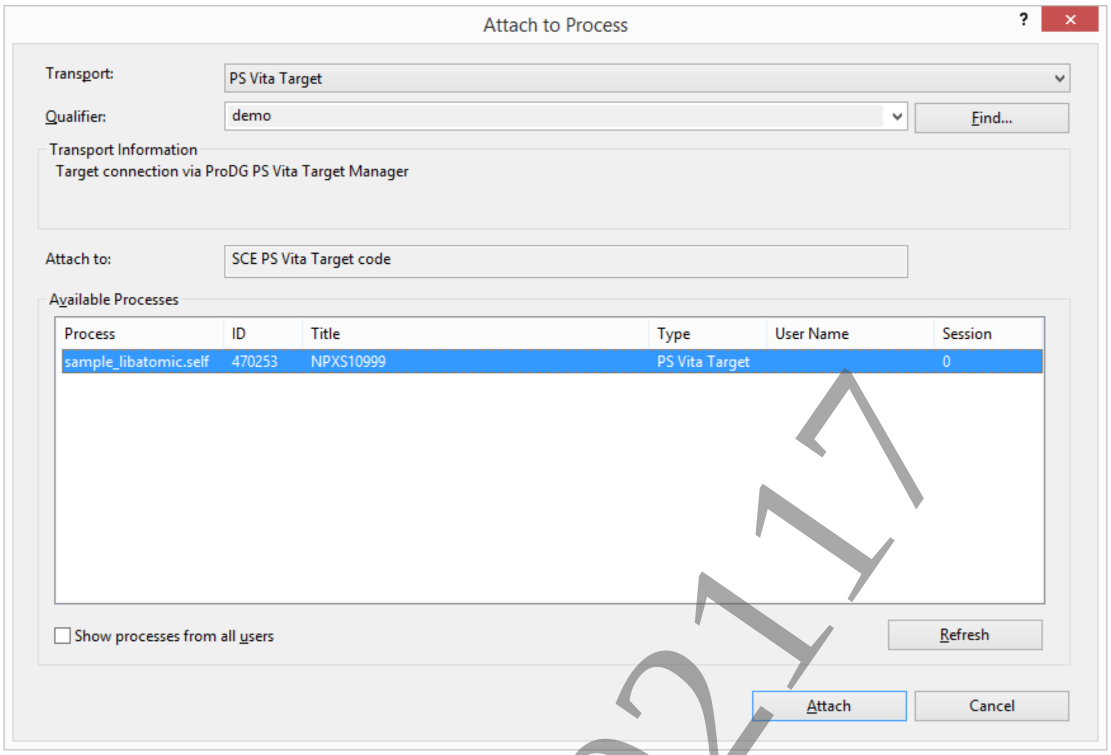
(4) Execute sample_libatomic.self Normally

Select **Debug** > **Start Without Debugging**. This will load the executable file (sample_libatomic.self) onto the DevKit and execution will start immediately.

(5) Attach a Running Process and Perform Debugging

Select **Debug** > **Attach to Process**. The **Attach to Process** dialog appears.

Figure 20 Attach to Process Dialog



Transport: PS Vita Target
Qualifier: demo
Transport Information: Target connection via ProDG PS Vita Target Manager
Attach to: SCE PS Vita Target code

Process	ID	Title	Type	User Name	Session
sample_libatomic.self	470253	NPXS10999	PS Vita Target		0

In the **Attach to Process** dialog box:

(1) In the **Transport** combo box, select **PS Vita Target**.

(2) Click **Browse** button, ensure the default DevKit is selected, and then click **OK** button.

(3) In the **Available Processes**, select the process, and then click **Attach** button.

(4) Select **Debug** > **Break All**.

(5) Step through the code.

(6) Save a Core Dump File

Perform the following procedure to save a core dump file.

(1) Select **Debug** > **Save SCE Process Dump As**.

(2) In the **Save as type** list in the **Save** dialog, select a mini dump or full dump, then click on the **Save** button.

This will save a core dump file to the specified location.

Once the core dump file is correctly created, stop debugging by selecting **Debug** > **Stop Debugging**.

(7) Load the Sample Core Dump File and Perform Debugging

By loading the saved core dump file to Visual Studio, it will be possible to resume debugging from the point where the core dump file was saved.

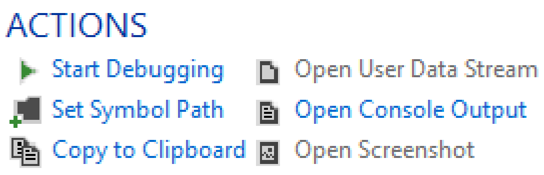
Here, a core dump file provided as a sample is loaded. Open the following core dump file using Windows Explorer.

```
%SCE_PSP2_SAMPLE_DIR%\sample_code\system\api_libatomic\sample_libatomic\PSP2
_Debug\sample_libatomic.psp2dmp
```

This will load the sample core dump file to Visual Studio.

After a core dump file is normally loaded, it will be possible to start debugging by selecting **ACTIONS** > **Start Debugging**.

Figure 21 CoreDump ACTIONS Menu



Note that when a sample core dump file is loaded, execution will stop after debugging starts due to a kernel exception. At this time, a message box will appear. Click on **Break**.

Application Performance Analysis

This tutorial explains the procedure for application performance analysis.

A complete set of files for this tutorial is available at the following location:

```
%SCE_PSP2_SAMPLE_DIR%\sample_code\developer_tools\api_libperf\
```

Note
For details on application performance analysis, refer to the "Razor User's Guide" document.

An outline of the execution procedure for this tutorial is as follows.

(1) Open the Sample Solution
(2) Open the main.c File and Check the Processing in Each Function
(3) Build the Solution
(4) Load the razor.self onto the DevKit
(5) Execute CPU Trace
(6) Display the CPU Trace Result Summary, Thread Information, and Function Information

(1) Open the Sample Solution

Open the following Visual Studio solution.

```
%SCE_PSP2_SAMPLE_DIR%\sample_code\developer_tools\api_libperf\api_libperf.sln
```

(2) Open the main.c File and Check the Processing in Each Function

In **Solution Explorer** pane, expand the **Razor** project, expand the **Source Files** folder, and then double-click **main.c**. This will display the main.c source code in the main window.

In order to understand the CPU trace results shown in step (6), confirm the processing in the following functions in main.c.

`main()` Function

The processing in the `main()` function is as follows.

- Calls the `init()` function.
- Calls the `basic()` function in a continuous loop.
- Calls the `shutdown()` function.

`init()` Function

The processing in the `init()` function is as follows.

- Loads the libperf module by calling `sceSysmoduleLoadModule()`.
- Resets all performance counters by calling `scePerfArmPmonReset()`.

- Selects six performance counters (two counters are repeated for simplicity) by calling `scePerfArmPmonSelectEvent()`.
- Starts measuring performance by calling `scePerfArmPmonStart()`.

Note
For more information on the performance counters, refer to "ARM Architecture Reference Manual ARMv7-A and ARMv7-R edition" document.

`basic()` Function

The processing in the `basic()` function is as follows..

- Pushes a user marker onto a thread-specific task by calling `sceRazorCpuPushMarkerWithHud()`.
- Increments the software increment performance counter once a frame for the first 16 frames by calling `scePerfArmPmonSoftwareIncrement()`.
- Increments the ISB instructions performance counter twice a frame for the first 16 frames by calling `__builtin_isb()`.
- Delays for half of the frame time by calling `sceKernelDelayThread()`.
- Pops the user marker by calling `sceRazorCpuPopMarker()`.
- Delays again for half of the frame time by calling `sceKernelDelayThread()`.
- Operates the synchronization point during capture stop using `sceRazorCpuSync()`.

`shutdown()` Function

The processing in the `shutdown()` function is as follows.

- Stops measuring performance by calling `scePerfArmPmonStop()`.
- Unloads the libperf module by calling `sceSysmoduleUnloadModule()`.

Note
Normally, the `shutdown()` function will not be reached because the sample runs in an infinite loop.

(3) Build the Solution

Select **Build** > **Build Solution**.
When the build is successful, the executable file razor.self is created.

(4) Load the razor.self onto the DevKit

Use the following procedure to load the executable file (razor.self) onto the DevKit.

(1) In **Solution Explorer** pane, right-click the **razor** project, and then click **Set as Startup Project** in the context menu.
(2) Select **RAZOR** > **Capture** > **Load Executable**.
(3) In the **Load Executable** dialog, select the **razor.self** file, and then click **Open** button.

(5) Execute CPU Trace

Use the following procedure to perform a CPU trace for 5 seconds.

(1) Select **RAZOR** > **Capture** > **CPU Trace**.
(2) After 5 seconds, in the **Capture** message box, click **Stop** button.

This will display the CPU trace results as a new tabbed document named "PS Vita Snapshot".

(6) Display the CPU Trace Result Summary, Thread Information, and Function Information

Immediately after the CPU trace stops, the CPU trace result summary will display. Various types of information can be displayed by selecting a view in the **Current View** combo box.

The information that can be displayed in CPU trace results is as follows.

Timeline

Displays thread activity as blue event bars for frames in the trace and idle threads as gray event bars. Processes, thread scheduling, system calls, and user marker event bars are displayed in this view.

Function List

Displays information for all functions based on the debug information of the currently selected module.

Hierarchical Functions

Displays inclusive and exclusive function timings, the number of calls, system calls, and interrupts for each function in your game.

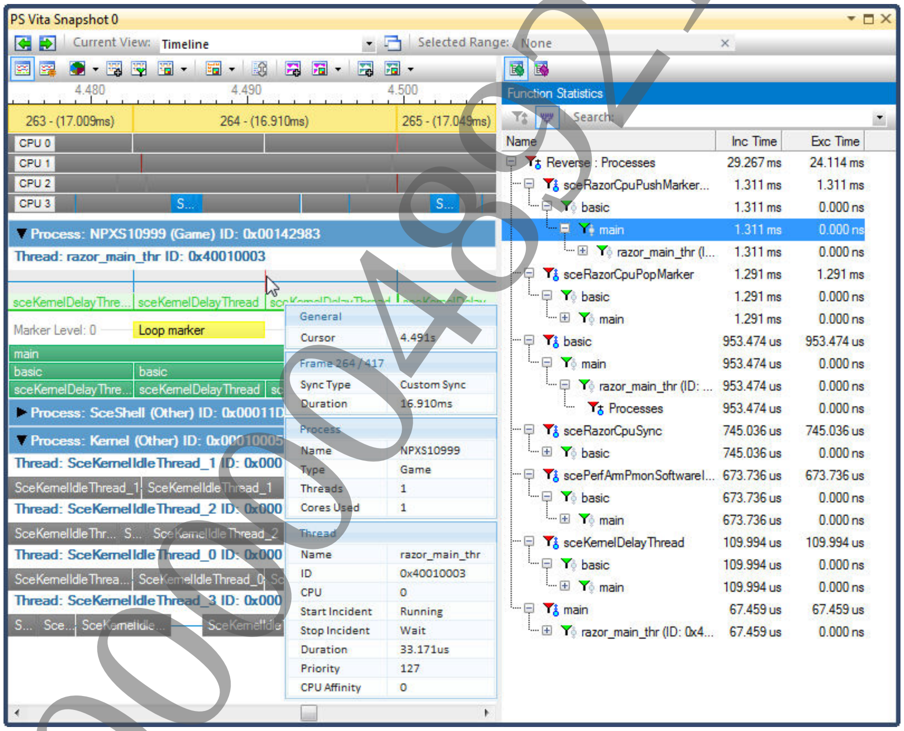
Note
You must perform a trace with the **Program Flow Trace** profiling mode enabled in the Razor options to obtain the **Hierarchical Functions** view.

Hierarchical User Markers

Displays inclusive and exclusive time for user marker events, and also the number of calls.

CPU Timeline

Displays thread activity for each process, and then for each CPU core in the trace.


PS Vita Snapshot 0
Current View: Timeline
Selected Range: None
4.480
4.490
4.500
263 - (17.009ms)
264 - (16.910ms)
265 - (17.049ms)
CPU 0
CPU 1
CPU 2
CPU 3
Process: NPXS10999 (Game) ID: 0x00142983
Thread: razor_main_thr ID: 0x40010003
Marker Level: 0
Loop marker
Process: SceShell (Other) ID: 0x00011D
Process: Kernel (Other) ID: 0x00010005
Thread: SceKernelIdleThread_1 ID: 0x000
Thread: SceKernelIdleThread_2 ID: 0x000
Thread: SceKernelIdleThread_0 ID: 0x000
Thread: SceKernelIdleThread_3 ID: 0x000
Function Statistics
Search:

General	
Cursor	4.491s
Frame 264 / 417	
Sync Type	Custom Sync
Duration	16.910ms
Process	
Name	NPXS10999
Type	Game
Threads	1
Cores Used	1
Thread	
Name	razor_main_thr
ID	0x40010003
CPU	0
Start Incident	Running
Stop Incident	Wait
Duration	33.171us
Priority	127
CPU Affinity	0

Name	Inc Time	Exc Time
Reverse : Processes	29.267 ms	24.114 ms
sceRazorCpuPushMarker...	1.311 ms	1.311 ms
basic	1.311 ms	0.000 ns
main	1.311 ms	0.000 ns
razor_main_thr (I...	1.311 ms	0.000 ns
sceRazorCpuPopMarker	1.291 ms	1.291 ms
basic	1.291 ms	0.000 ns
main	1.291 ms	0.000 ns
basic	953.474 us	953.474 us
main	953.474 us	0.000 ns
razor_main_thr (ID: ...	953.474 us	0.000 ns
Processes	953.474 us	0.000 ns
sceRazorCpuSync	745.036 us	745.036 us
basic	745.036 us	0.000 ns
scePerfArmPmonSoftwareI...	673.736 us	673.736 us
basic	673.736 us	0.000 ns
main	673.736 us	0.000 ns
sceKernelDelayThread	109.994 us	109.994 us
basic	109.994 us	0.000 ns
main	109.994 us	0.000 ns
main	67.459 us	67.459 us
razor_main_thr (ID: 0x4...	67.459 us	0.000 ns

Note
In addition to CPU traces, Razor enables you to obtain and analyze GPU captures, GPU traces, GPU live metrics, and also CPU and GPU HUD information.
All graphics samples in folder `%SCE_PSP2_SAMPLE_DIR%\sample_code\graphics\api_libgxm` of the SDK include Razor GPU capture and GPU HUD feature.
The gpu_live_metrics_basic sample in folder `%SCE_PSP2_SAMPLE_DIR%\sample_code\graphics\api_razor\gpu_live_trace_metrics` of the SDK includes GPU live metrics.
The tutorial_physics_effects_high_level sample in folder `%SCE_PSP2_SAMPLE_DIR%\sample_code\engines\tutorial_physics_effects_high_level` of the SDK includes CPU HUD.
For details, refer to the "Razor User's Guide".

Appendix A: Using Offline Installation

This chapter explains the offline installation features of the SDK Manager.

SDK Manager has a feature to create the offline installer package. By using the offline installer, it is possible to set up the SDK on development host computers that cannot access the Internet.

The offline installation features comprise the following..

- SDK distribution installer
- Download files

Standalone Installer (Offline Installer)

The Standalone Installer generates an SDK distribution kit for installing offline.

The downloaded files will be packaged in the destination directory specified on the package selection screen, as a complete offline installer solution.

The offline installation package is created with the following layout:

offline_installer_pkg	
`SDK_Manager.exe`	The executable file of SDK Manager
`+ \InstallFiles`	The files for SDK Manager to install
`ActionScript.xml`	The configuration files of SDK Manager
`Manifest.imf`	

By using the offline installer of the created offline_installer_pkg, each component can be distributed without having to connect to the Internet. Execute the SDK_Manager.exe file to install SDK Manager and set up the contents offline.

Download Files

File downloads are performed in **Download Files** without installing components. Each component is downloaded to the respective folders for the package names displayed in SDK Manager. This is an ideal solution for cases where component installation management is required, such as when using source management for distribution of SDKs within a team.

These components consist of individual file formats, ZIP formats, or MSI/exe format installers.

SDK Manager does not perform the installation steps or setup process in this mode. Configuration for environment variables, etc. must be performed manually.

For details on SDK layout settings, environment variables, and required installation settings, refer to the "Verifying the Setup" section in Chapter 2: "Installing Development Software".

Appendix B: Other System Update Methods

System Update Using the Setting Application of the System Software of the DevKit

System update can be performed using the setting application of the system software.

The system update methods using the settings are as follows.

- Update via Network Using Wi-Fi (DevKit/TestKit)
- Update via Network by Connecting to PlayStation®3 (DevKit/TestKit)
- Update via Network by Connecting to a PC (DevKit/TestKit)
- Update Using Storage Media (Update Using Memory Card) (DevKit/TestKit)
- Update Using MTP (Update by Referring to a File on a PC Through CMA) (DevKit/TestKit)

Preparations for Update Performed via Network

(1) Store the Update Data on the HTTP Server
Create an arbitrary directory on the HTTP server and store the update data in this folder.
The update data is "`%SCE_ROOT_DIR%\PSP2\System Update Files\PSP2UPDAT-<VERSION>(-<MODEL NAME>).PUP`". The filename, including the extension, is arbitrary - it can be changed to a name that is easy to manage. However, make sure that the pathname, including the server URL, does not exceed 1024 characters.

(2) Create an Update List File
An update list file is a file in which the path of an update data that is stored on the HTTP server is written in a designated format. Refer to the "Update List File Format" section below and create the update list file. Store this file in an arbitrary directory on the HTTP server. It can be in the same directory as the update data, or in a different one. The filename, including the extension, is arbitrary. However, make sure that the pathname, including the server URL, does not exceed 511 characters.

(3) Set the URL of the Update File List to the DevKit or the TestKit
Start a setup of Neighborhood or start the Settings application of the system software, and open ★**Debug Settings** > **System Update** > **Update Server URL**. Enter the URL (including the update list file), and click the **Enter** button to save it.
If updating through connection to the PC, set a URL to reference update list files from the PC.

Update via Network Using Wi-Fi (DevKit/TestKit)

The update data is stored on the HTTP server on the local network, and system updates can be done via a network. This method is useful when using numerous DevKits and TestKits, because the update data can be managed all together. However, the DevKit system software must be updated to 0.940 or later. Like the method of the "Update Using Storage Media (Update Using Memory Card) (DevKit/TestKit)" item, it is possible to perform update with selecting the applicable data from multiple update files by using a ★ **Debug Settings** setup.

In order to use this feature, it is necessary to set up a network beforehand. Refer to the "Network" section in the "Setting Functions" chapter in the "System Software Overview" document for the procedure of network configuration.

Execute the System Update Using Wi-Fi

After the completion of the procedure described in the above the "Preparations for Update Performed via Network" section, select **System Update** > **Update Using Wi-Fi** from the Settings application of the system software. The update data stored in the HTTP server is installed. When the ★**Debug Settings > System Update** > **Show PUP List** is set as **ON** by a settings of Neighborhood or from the Settings application of the system software, a list of update data stored on the HTTP server will be displayed onscreen, and the selected update data will be installed.

Update via Network by Connecting to PlayStation®3 (DevKit/TestKit)

This method carries out the update by connecting PlayStation®3 to the multi-use port of DevKit or TestKit (PTEL-1000 series) or to the USB port of TestKit (PTEL-2000 series) using a USB cable.

To be able to use **Update by Connecting to a PS3™ System**, the DevKit system software must be updated to 0.990 or later.

In addition, the version of the system software of PlayStation®3 to be connected to must be updated beforehand to 4.00 or later. Also, the connected PlayStation®3 must be set up so as to enable access to the HTTP server where the update list files are placed.

Furthermore, it is required to log in to PlayStation®3 using a PlayStation®3 account and then connect a USB cable. On the PlayStation®3 side, the user is prompted to sign in with the Sony Entertainment Network account after the connection is established; however, such restriction is not applied to the DevKit or the TestKit. Subsequent processing can be performed with the sign-in screen being displayed on the PlayStation®3 side.

Execute the System Update by Connecting to PlayStation®3

After the completion of the procedure described in the above the "Preparations for Update Performed via Network" section, select **System Update** > **Update by Connecting to a PS3™ System** from the Settings application of the system software.

Update via Network by Connecting to a PC (DevKit/TestKit)

This method carries out the update by connecting a commonly used PC to the multi-use port of DevKit or TestKit (PTEL-1000 series) or to the USB port of TestKit (PTEL-2000 series) using a USB cable.

To be able to use **Update by Connecting to a PC**, the DevKit system software must be updated to 0.990 or later.

In addition, install the "Content Manager Assistant for PlayStation®Vita DevKit (`%SCE_ROOT_DIR%\PSP2\Services\Content Manager Assistant for PlayStation(R)Vita DevKit\installer`)" on the PC to be connected in advance, and enable connection with the PC.

Also, the connected PC must be set up so as to enable access to the HTTP server where the update list files are placed.

Execute the System Update Connecting to a PC

After the completion of the procedure described in the above the "Preparations for Update Performed via Network" section, select **System Update** > **Update by Connecting to a PC** from the Settings application of the system software.

Update Using Storage Media (Update Using Memory Card) (DevKit/TestKit)

In this method, update is performed by referring to an update data stored in a memory card. This method can be used by saving "`%SCE_ROOT_DIR%\PSP2\System Update Files\PSP2UPDAT-<VERSION>(-<MODEL NAME>).PUP`" as `ux0:data/PSP2/UPDATE/PSP2UPDAT.PUP` to a memory card in advance.

In addition, by creating files with arbitrary names under `ux0:data/PSP2/UPDATE/SEARCH/`, multiple update files can be stored. (Do not change the extension. Continue to use PUP.) When storing multiple update files, set ★**Debug Settings** > **System Update** > **Show PUP List** to **On** from the Settings application of the system software.

For the DevKit, the reference destination of the update file can also be set to refer to File Serving Directory on the development host computer. For details, refer to the "System Software Overview" document.

Use the DevKit to write PUP into the memory card.

Execute Update Using Storage Media

Insert the memory card in the memory card slot of the DevKit or the TestKit, and then, select **System Update** > **Update Using Storage Media** from the Settings application of the system software, and perform update according to the instructions displayed on the screen. Refer to the "How to Use Memory Cards" section in the chapter 3 "Connecting, System Update, and Checking Operation" on how to copy files.

Update Using MTP (Update by Referring to a File on a PC Through CMA) (DevKit/TestKit)

With CMA, perform update referring to the update data stored on the PC side. This is the simplest method for updating the TestKit.

Connect the DevKit or TestKit to a PC on which CMA has already been installed and then save "`%SCE_ROOT_DIR%\PSP2\System Update Files\PSP2UPDAT-<VERSION>(-<MODEL NAME>).PUP`" as "`My Documents\PS Vita\package\UPDATE\PSP2UPDAT.PUP`" ("My Documents\PS Vita" is a path defined with **Applications/Backup Files** of CMA settings).

Connect the PC to the multi-use port of DevKit or TestKit (PTEL-1000 series) or to the USB port of TestKit (PTEL-2000 series) using a USB cable.

After checking that the connection using CMA has been successfully established, select **System Update > Update Using MTP** from the Settings application of the system software, and perform update according to the instructions displayed on the screen.

Although PSP2UPDAT.PKG is also supported to maintain compatibility, PUP will be prioritized when PSP2UPDAT.PUP exists in the same folder. If neither file can be found, "**Cannot find the file.**" system software error will return; check the filename and path. PSP2UPDAT.PUP is supported in system software 1.800 and later. For system softwares earlier than 1.800, use PSP2UPDAT.PKG.

Update List File Format

An update list file describes by XML. There is sample as follows. The underlined items are items of which descriptions need to be changed accordance with your development environment.

```
<?xml version="1.0" encoding="UTF-8"?>
<update_data_list>
  <region id="257"> <!-- Development Kit -->
    <version system_version="0.940">
      <update_data>
        <image>http://192.168.0.20/images/PSP2UPDAT-0940.PUP</image>
      </update_data>
    </version>
    <version system_version="0.931">
      <update_data>
        <image>http://192.168.0.20/images/PSP2UPDAT-0931.PUP</image>
      </update_data>
    </version>
    <version system_version="0.930">
      <update_data>
        <image>http://192.168.0.20/images/PSP2UPDAT-0930.PUP</image>
      </update_data>
    </version>
  </region>
  <region id="258"> <!-- Testing Kit -->
    <version system_version="1.600">
      <update_data>
        <image>http://192.168.0.20/images/PSP2UPDAT-1600-TestKit.PUP</image>
      </update_data>
    </version>
  </region>
</update_data_list>
```

In this example, there are three update data files (0.940, 0.931, 0.930) for the DevKit and one update data file (1.600) for the TestKit. Each update file is stored to the HTTP server (IP address: 192.168.0.20) in the images directory under the DocumentRoot directory. They are named `PSP2UPDAT-<VERSION>.PUP`.

The form of an update list file is as follows.

Elements / Attributes	Description
`<update_data_list>`	This is a fixed route element. One is certainly required. A value is not described.
`<region>`	Only one is described as a child element of `<update_data_list>` element. The following `region@id` attribute is required.
`region@id`	This is an attribute essential in a `<region>` element. Describe the following values: For Development Kit: "257" For Testing Kit: "258"
`<version>`	The child element of the `<region>` element can describe to 20 pieces. The `<version>` element is described for every version of the system update data. When ★**Debug Settings > System Update > Show PUP List** of the Neighborhood setting or Settings application of the system software is set to **ON**, the list of update data files will be displayed to 20 pieces. When **Show PUP List** is set to **OFF**, only the top `<version>` element is displayed.

Elements / Attributes	Description
`version@system_version`	Version number to be listed onscreen when the update is executed on the DevKit. The format is one numeric character + period + four numeric characters.
`<update_data>`	Only one is described as a child element of `<version>` element.
`<image>`	Only one is described as a child element of `<update_data>` element. URL of the update data. Enter the full path beginning with "http://". The length of URL can be up to 1024 characters.

When two or more "Only one is described" elements have been described, only the first element is evaluated. The maximum size of the file itself is 65,535 bytes. The description of those other than a predetermined element and an attribute is disregarded.

Update List File Parsing Errors

If there are any errors in the update list files, an error dialog stating "An error has occurred. (xxxxxxxx)" appears to the timing which performed system update on PlayStation®Vita. The error code displayed in an error dialog has the following meanings.

Error code	Description
C3-10691-7	The number of `<version>` elements is over 20.
C3-10692-8	The number of `<update_data>` elements is over 256.
C3-10693-9	URL described of `<image>` is over 1024 characters.
C3-10694-0	`region@id` attribute does not exist.
C3-10695-1	`version@system_version` attribute does not exist. The format of the version information described for the `version@system_version` attribute is inaccurate.
C3-10697-3	Route element of `<update_data_list>` does not exist.
C3-10698-4	`<region>` element does not exist.
C3-10699-5	`<version>` element does not exist.
C3-10700-8	`<update_data>` element does not exist.
C3-10701-9	The unknown error occurred.

The network-related error is not included in the above-mentioned table. Refer to the "Network Overview" document.

Appendix C: Method to Manually Install the USB Driver

When the USB port (mini-B) for development of DevKit is connected to the development host computer by a USB cable, use the following method to update the USB driver if DevKit is not recognized.

If the USB driver is installed by using SDK Manager, you do not need to manually install the driver. However, if DevKit is not recognized for any reason, follow the procedure below to install the driver.

When the "Found New Hardware" Dialog Box or "Found New Hardware Wizard" Dialog Box Are Displayed

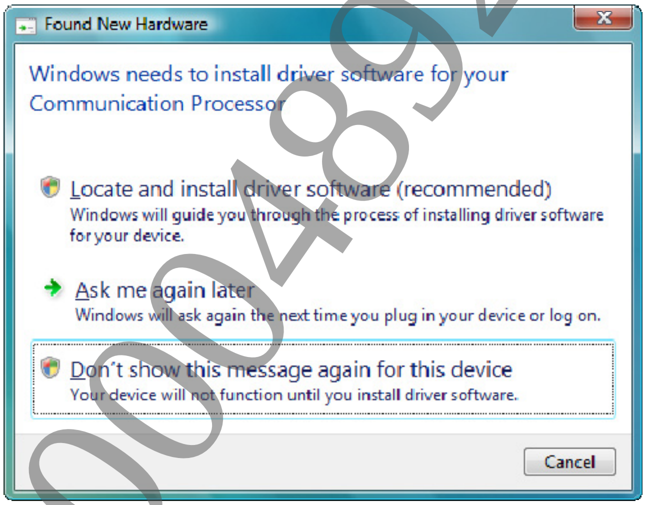
When the "Found New Hardware" dialog box or the "Found New Hardware Wizard" dialog box are displayed, you can use the method described below to install/update the DevKit's USB driver (this explanation assumes the use of Windows 7)

(1) Download and install the USB driver and SDK using SDK Manager.
Normally, the USB driver is installed with this procedure.

(2) Connect the DevKit to the development host computer by using the supplied USB cable.
Provided that the USB driver is installed correctly, it will be possible to use it in this state.

(3) If the **Found New Hardware** dialog box appears, select **Don't show this message again for this device**.

Figure 22 "Found New Hardware" Dialog Box


Found New Hardware

Windows needs to install driver software for your Communication Processor

Locate and install driver software (recommended)
Windows will guide you through the process of installing driver software for your device.

Ask me again later
Windows will ask again the next time you plug in your device or log on.

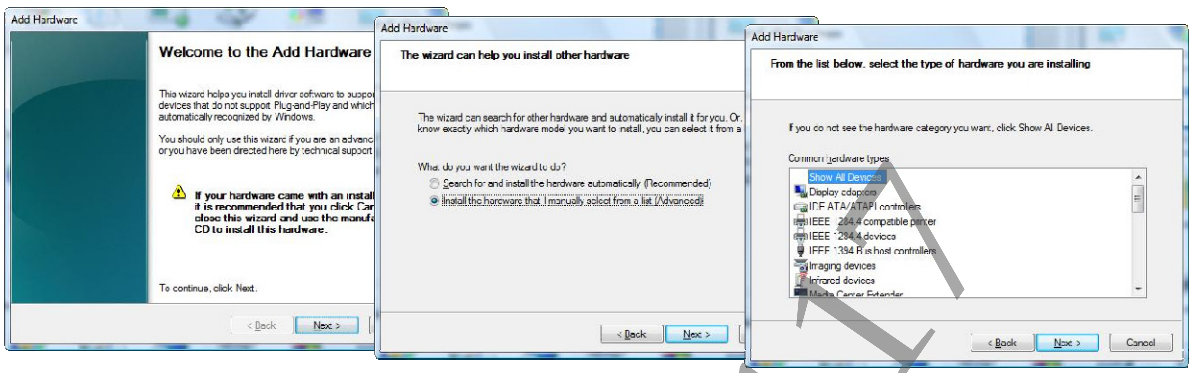
Don't show this message again for this device
Your device will not function until you install driver software.

(4) In Control Panel, open **Add Hardware**.

(5) The Add Hardware wizard appears.

5-1. Click **Next**.
5-2. Click **Install the hardware that I manually select from a list (Advanced)**, and then click **Next**.
5-3. Click **Show All Devices**, and then click **Next**.

Figure 23 Add Hardware Wizard



Add Hardware

Welcome to the Add Hardware

This wizard helps you install driver software to suppo
devices that do not support Plug-and-Play and which
automatically recognized by Windows.

You should only use this wizard if you are an advanc
or you have been directed here by technical support

If your hardware came with an install
it is recommended that you click Car
close this wizard and use the manufa
CD to install this hardware.

To continue, click Next.

< Back | Next >

Add Hardware

The wizard can help you install other hardware

The wizard can search for other hardware and automatically install it for you. Or
know exactly which hardware model you want to install, you can select it from a

What do you want the wizard to do?

Search for and install the hardware automatically (Recommended)
[✓ Install the hardware that I manually select from a list (Advanced)]

< Back | Next >

Add Hardware

From the list below, select the type of hardware you are installing

If you do not see the hardware category you want, click Show All Devices.

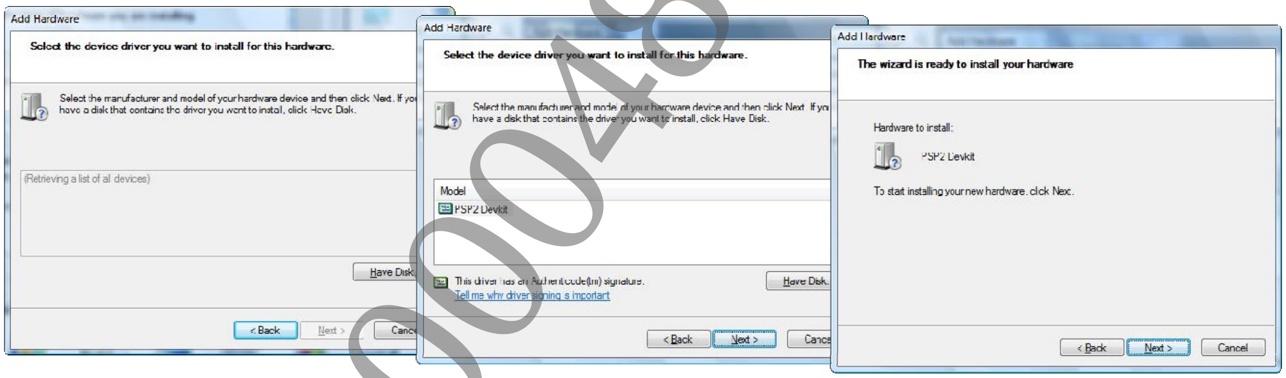
Common hardware types:

[✓ Show All Devices]
Display adapters
IDE ATA/ATAPI controllers
IEEE 1284.4 compatible printer
IEEE 1284.4 devices
IEEE 1394 Bus host controllers
Imaging devices
Infrared devices
Media Center Extender

< Back | Next > | Cancel

5-4. Click **Have Disk**.
5-5. Select the file at the following location (*** depends on the operating system), and then click **Next**.
`"%SCE_SDK_ROOT%\PSP2\Services\USB Driver for PlayStation(R)Vita DevKit\driver\***\sce_psp2_usb.inf"`
5-6. Click **Next** to start the installation.

Figure 24 Add Hardware Wizard - Installation



Add Hardware

Select the device driver you want to install for this hardware.

Select the manufacturer and model of your hardware device and then click Next. If yo
have a disk that contains the driver you want to install, click Have Disk.

(Retrieving a list of all devices)

Have Disk

< Back | Next > | Canc

Add Hardware

Select the device driver you want to install for this hardware.

Select the manufacturer and model of your hardware device and then click Next. If yo
have a disk that contains the driver you want to install, click Have Disk.

Model
PSP2 Devkit

This driver has an Authenticode(tm) signature.
Tell me why driver signing is important

Have Disk

< Back | Next > | Canc

Add Hardware

The wizard is ready to install your hardware

Hardware to install:

PSP2 Devkit

To start installing your new hardware, click Next.

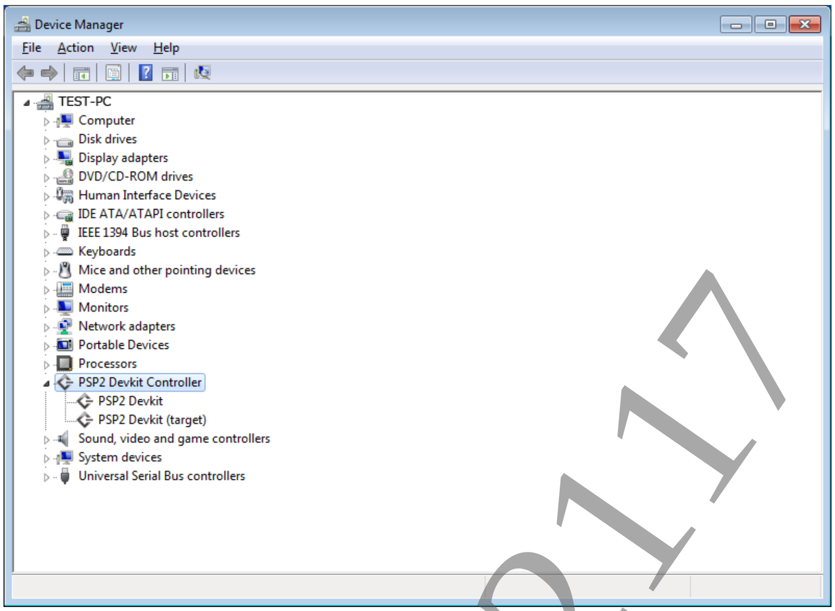
< Back | Next > | Cancel

Note
The USB driver can also be installed in Device Manager by using the PSP2 DevKit Controller property.

(6) Open the Device Manager (devmgmt.msc) and check whether installation has been performed correctly.

(7) Check that **PSP2 DevKit Controller** > **PSP2 DevKit** and **PSP2 DevKit (target)** are displayed normally. If there are any problems, a warning mark will be displayed. Try **Update Driver...**.

Figure 25 Device Manager



If there is a problem, a warning mark is displayed.

Update the USB Driver

Note
In this work, the reboot of a computer is needed.

(1) Open the Device Manager (devmgmt.msc).

(2) Right-click **PSP2 DevKit** under **PSP2 DevKit Controller**. Then, select **Update Driver...**.

(3) As in the Step 2, right-click **PSP2 DevKit (target)** under **PSP2 DevKit Controller**. Then, select **Update Driver...**.

(4) Reboot the computer.

Uninstall the USB Driver

Note
In this work, the reboot of a computer is needed.

(1) Open the Device Manager (devmgmt.msc).

(2) Right-click **PSP2 DevKit** under **PSP2 DevKit Controller**. Then, select **Uninstall**.

(3) As in the Step 2, right-click **PSP2 DevKit (target)** under **PSP2 DevKit Controller**. Then, select **Uninstall**.

(4) Reboot the computer.

Appendix D: TestKit Setup

Explanations of TestKit setup are summarized below. For detailed explanations, refer to each chapter.

Overview

Because the TestKit differs from the DevKit in that a connection via the CP is not made with the development host computer, the Neighborhood and psp2ctrl features cannot be used.

Therefore, a memory card or CMA is used for updates, activation, and pkg file installation. Note that the CMA mentioned here does not refer to Content Manager Assistant for PlayStation® but Content Manager Assistant for PlayStation®Vita DevKit.

When sending a file to the TestKit using a memory card, use the DevKit to write the file into the memory card, making use of the File System Mapping feature of Neighborhood DevKit.

Connection Method

Similarly to the case of the retail unit, complete the network settings in advance and adjust the clock.

When using the memory card to perform an update or activation, or to install a pkg file, there is no need for TestKit to be connected to the development host computer.

When using the CMA to copy the update file, activation file, or pkg file, connect the PC - onto which the CMA is installed - to the multi-use port of TestKit (PTEL-1000 series) or to the USB port of TestKit (PTEL-2000 series) using a USB cable.

System Software Update

For updates of the system software of the TestKit, use either Wi-Fi or a memory card, making use of the system software update file (PUP) for the TestKit.

Use the following methods described in the "Appendix B: Other System Update Methods" chapter of this document.

- Update via Network Using Wi-Fi (DevKit/TestKit)
- Update Using Storage Media (Update Using Memory Card) (DevKit/TestKit)
- Update Using MTP (Update by Referring to a File on a PC Through CMA) (DevKit/TestKit)

Activation

Activation using CMA or a memory card can be executed with the TestKit.

For details on activation using a memory card, refer to the "Updating the Expiry Time of DevKit and TestKit Using CMA (Activate by Connecting PC) (DevKit/TestKit)" section or to the "DevKit/TestKit Activation User's Guide" document.

Installing a pkg File

Use CMA or a memory card to install pkg files.

For details, refer to the "Appendix 1: How to Install Packages" chapter in the "Application Development Process Overview" document.

Appendix E: Setup of the Development Environment for PlayStation®TV

This chapter explains the method to set up the development environment for PlayStation®TV.

Overview

The basic development environment is the same as PlayStation®Vita. Please have completed the setup of the development environment for PlayStation®Vita using DevKit (described earlier).

The following is an explanation on how to change the development environment for PlayStation®Vita to the development environment for PlayStation®TV.

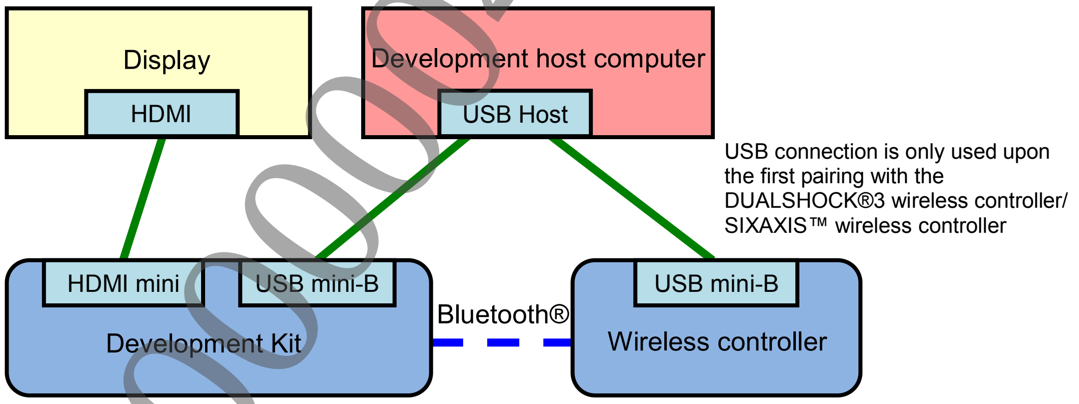
Development Environment

- Development host computer (Windows PC)
- DevKit: PDEL-1000 series
- Separately sold wireless controller
  SIXAXIS™ wireless controller, DUALSHOCK®3 wireless controller or the DUALSHOCK®4 wireless controller
- Display (input-enabled HDMI) (not a requirement)

Note
Wherever a distinction is not required in this document, the term "wireless controller" will be used in reference to the SIXAXIS™ wireless controller, the DUALSHOCK®3 wireless controller and the DUALSHOCK®4 wireless controller.

The system configuration is shown in the following diagram:

Figure 26 System Configuration


Display
HDMI
Development host computer
USB Host
USB connection is only used upon the first pairing with the DUALSHOCK®3 wireless controller/ SIXAXIS™ wireless controller
HDMI mini
USB mini-B
Development Kit
Bluetooth®
USB mini-B
Wireless controller

Development Tools

The following tools, packaged with the SDK, are used.

- Wireless controller driver (SCEWirelessControllerDriverv2.50.2.0.msi or later)
  The wireless controller driver is only required when using a DUALSHOCK®3 wireless controller or SIXAXIS™ wireless controller.
- Neighborhood for PlayStation®Vita

The wireless controller driver is provided to be installed onto a development host computer for the pairing of DevKit and the DUALSHOCK®3 wireless controller/SIXAXIS™ wireless controller. This wireless controller driver is an expanded version of the wireless controller driver included in libpad for Windows of the PlayStation®3 SDK (expanded to support pairing with DevKit) with an installer added to it. If you are already using the wireless controller driver of the PlayStation®3 SDK, it is necessary to update to the wireless controller device included in this PlayStation®Vita SDK.

Setup Method

(1) Connect DevKit

Use a commercially available HDMI cable and connect the HDMI mini port of DevKit with the HDMI input port of the display. Also, in the same manner as with the development environment for PlayStation®Vita, connect the USB port (mini-B) for development of DevKit with the development host computer using a USB cable.

(2) Switch PS TV Emulation

Set ★**Debug Settings** > **Boot Parameters** > **PS TV Emulation** > **On** from Neighborhood or from the Settings application of the system software.

(3) Connect a DUALSHOCK®3 Wireless Controller or SIXAXIS™ Wireless Controller

To connect DevKit with a DUALSHOCK®3 wireless controller or SIXAXIS™ wireless controller using Bluetooth®, carry out pairing using the development host computer, as follows.

(1) Install the wireless controller driver packaged with the SDK onto the development host computer. On the screen to select a package of the SDK Manger, mark the checkbox for **External Tools > Wireless Controller Driver - 'DUALSHOCK 3' - X.X.X.X** and install the SDK. The installer of the wireless controller driver will be placed in the following folder.
%SCE_ROOT_DIR%\Common\External Tools\Wireless Controller Driver
Double-click on SCEWirelessControllerDrivervX.X.X.X.msi in the folder and install as instructed onscreen.

(2) Connect the wireless controller to the development host computer with a USB cable.

(3) Press the **PS** button of the wireless controller.

(4) Right-click the target on Neighborhood and select **Pair DUALSHOCK(R)3 Controller** from the displayed context menu. **Pair DUALSHOCK(R)3 Controller** can only be used when **PS TV Emulation** is set to **On**.

(5) Remove the USB cable from the wireless controller. Check that DevKit can be operated from the wireless controller.

Notes

In a state where pairing of the wireless controller has already been performed with another device (PlayStation®3, for example) and a connection is established (PlayStation®3 is turned on), it may not be possible for DevKit to use the wireless controller. In this case, turn the power of the paired device off; it will become possible for DevKit to use the wireless controller.

(4) Connect a DUALSHOCK®4 Wireless Controller

When connecting the DUALSHOCK®4 wireless controller to DevKit, carry out the following.

(1) In the context menu displayed by right-clicking on a target in Neighborhood, select **Pair DUALSHOCK(R)4 Controller**. **Pair DUALSHOCK(R)4 Controller** can only be used when **PS TV Emulation** is set to **On**.

(2) The DUALSHOCK®4 Pairing application will start in the DevKit, the "Keep pressing PS + SHARE buttons." message will be displayed, and the connection wait time count will start.
Before the count reaches 0, press and hold the SHARE button and PS button on a DUALSHOCK®4 wireless controller until the light bar flashes white.

(3) Usage is possible once the DUALSHOCK®4 wireless controller light bar is illuminated.

In addition, another DUALSHOCK®4 wireless controller can be connected with the following procedure.

(1) Select **Devices** > **Bluetooth® Devices** from the Settings application of the system software.

(2) Press and hold the SHARE button and PS button of the DUALSHOCK®4 wireless controller until the light bar flashes white.

(3) Select the **Wireless Controller** added to **Bluetooth® Devices**.

(4) A dialog to confirm the registration will be displayed; select **OK**.
When the light bar of the DUALSHOCK®4 wireless controller flashes red, The previously connected controller will be connected as Controller Number 1, and the other DUALSHOCK®4 wireless controller will be connected as Controller Number 2 (red).
Change the Controller Number as necessary from **Settings** > **Devices** > **Wireless Controller** > **Controller Number**.

Notes

When a DUALSHOCK®4 wireless controller is paired though Neighborhood, all device registrations for DUALSHOCK®4 wireless controllers not connected will be removed. If you do not want a device registration to be removed, press the PS button of the DUALSHOCK®4 wireless controller to connect it before performing pairing. To register it again after it has been deregistered, perform pairing again using the procedure in "Connect a DUALSHOCK®4 Wireless Controller".

Operation Method

When the setting is ★**Debug Settings** > **Boot Parameters** > **PS TV Emulation** > **On** of the system software Settings application, the touch panels of DevKit cannot be used. The operation of the system software can be performed using the wireless controller. Operation method in this case will be pursuant to the button operation of the PlayStation®Vita system.

Safe Mode

Use buttons on DevKit to operate upon safe mode startup and during safe mode.

Version Down of the System Software

When the Settings application of the DevKit system software is set as ★**Debug Settings** > **Boot Parameters** > **PS TV Emulation** > **On**, lowering the version to one less than system software 3.000 will return an error. When lowering the system software version to less than 3.000, set **PS TV Emulation Off**.

Restrictions

In a state where **PS TV Emulation** is **On**, do not lower the system software version to one less than 3.000.

In safe mode, even when **PS TV Emulation** is **On**, it is possible to lower the version to less than system software 3.000 without generating an error. When the system software is lowered to less than version 3.000 in this way with **PS TV Emulation > On**, the touch panels and buttons of DevKit may become inoperable.

In such a case, use the safe mode and update the system software to 3.000 or later, set **PS TV Emulation > Off**, and lower the system software version again.