Touch Service Overview

PlayStation®Vita Programmer Tool Runtime Library Release 3.50

Document serial number: 000004892117

Table of Contents

1 Library Overview

Scope of This Document

This document describes the Touch Service for reading input data from the touch panels. Because the Touch Service samples data of the touch panels at the VSYNC period, the program can properly obtain data in the Touch Service queued within the VSYNC period. This document also explains the handling when data cannot be sampled for two or more periods, and which obtainment method an application should select if multiple methods are available.

Note
The touch panel on the front of the PlayStation®Vita is referred to as the "screen (touchscreen)" since it includes the display screen, and the touch panel on the back is referred to as the "rear touch pad".
Since this document explains the methods for obtaining input data from touch panels, the "screen (touchscreen)" is referred to as the "front touch panel", the "rear touch pad" is referred to as the "rear touch panel", and the "screen (touchscreen)" and "rear touch pad" are both referred to as the general term "touch panel(s)".

Purpose and Features

The Touch Service is a service for reading the input data from the touch panels.

The major feature of the Touch Service is as follows.

- Obtaining data from the touch panels

Files

The following files are required to use the Touch Service.

Filename	Description
libSceTouch_stub.a	Stub library file
touch.h	Header file

Sample Programs

Sample programs are provided in the following directory as examples of programs that use the Touch Service.

sample_code/input_output_devices/api_touch/basic/

This sample shows the basic procedure for using the Touch Service.

sample_code/input_output_devices/api_touch/trace/

This sample traces the touch positions visually.

2 Usage Procedure

Basic Usage Procedure (If Calls Can Be Made Properly at the VSYNC Period)

The Touch Service performs sampling of the data of the touch panels at the VSYNC period.

Therefore, a program which calls the `sceTouchRead()` function at the VSYNC period will be capable of properly obtaining data which has been queued by the Touch Service.

(1) Initialization

The Touch Service runs as a default module. Explicit initialization is not necessary.

(2) Obtaining Touch Panel Information

Touch panel information is obtained with the `sceTouchGetPanelInfo()` function.

To obtain the information of the front touch panel, specify `SCE_TOUCH_PORT_FRONT` for the *`port`* argument.

To obtain the information of the rear touch panel, specify `SCE_TOUCH_PORT_BACK` for the *`port`* argument.

For details on the touch panel coordinate information, refer to the section "Coordinate System of Touch Data"

(3) Setting the Sampling Port

Using the `sceTouchSetSamplingState()` function, make the settings of the touch panel for which sampling is to be done with the Touch Service.

To enable the front touch panel, specify `SCE_TOUCH_PORT_FRONT` in the *`port`* argument and specify `SCE_TOUCH_SAMPLING_STATE_START` in the *`state`* argument.

To enable the rear touch panel, specify `SCE_TOUCH_PORT_BACK` in the *`port`* argument and specify `SCE_TOUCH_SAMPLING_STATE_START` in the *`state`* argument.

By default, neither the front touch panel nor the rear touch panel is enabled, so touch data cannot be obtained unless this setting is made.

(4) Obtaining Data from the Touch Service

Touch data can be obtained in blocking fashion by calling the `sceTouchRead()` function.

```
SceTouchData tdf, tdb;
int res;

// Obtain data from the front touch panel
res = sceTouchRead(SCE_TOUCH_PORT_FRONT, &tdf, 1);

// Obtain data from the rear touch panel
res = sceTouchRead(SCE_TOUCH_PORT_BACK, &tdb, 1);
```

When Calls may Be Made at Periods of Two VSYNCs or More

If processing delays occur in an application, then the `sceTouchRead()` function, which is essentially supposed to be called by an application at the VSYNC period, may end up not getting called for two or more periods.

Touch data loss may be a cause of unobtained touch gestures or misjudgment; therefore, all the data should be obtained without fail. The Touch Service holds 64 pieces of history information and provides a functionality as an API to obtain multiple data at one time, including unobtained data. Below is the procedure for obtaining multiple data in a collective manner.

(1) Initialization

The Touch Service runs as a default module. Explicit initialization is not necessary.

(2) Obtaining the Touch Panel Information

The touch panel information is obtained with the `sceTouchGetPanelInfo()` function.

To obtain the information of the front touch panel, specify `SCE_TOUCH_PORT_FRONT` for the *`port`* argument.

To obtain the information of the rear touch panel, specify `SCE_TOUCH_PORT_BACK` for the *`port`* argument.

For details on the touch panel coordinate information, refer to the section "Coordinate System of Touch Data."

(3) Setting the Sampling Port

Using the `sceTouchSetSamplingState()` function, make the settings of the touch panel for which sampling is to be done with the Touch Service.

To enable the front touch panel, specify `SCE_TOUCH_PORT_FRONT` in the *`port`* argument and specify `SCE_TOUCH_SAMPLING_STATE_START` in the *`state`* argument.

To enable the rear touch panel, specify `SCE_TOUCH_PORT_BACK` in the *`port`* argument and specify `SCE_TOUCH_SAMPLING_STATE_START` in the *`state`* argument.

By default, neither the front touch panel nor the rear touch panel is enabled, so touch data cannot be obtained unless this setting is made.

(4) Obtaining Data from the Touch Service

When calling the `sceTouchRead()` function, specify multiple buffers in the argument.

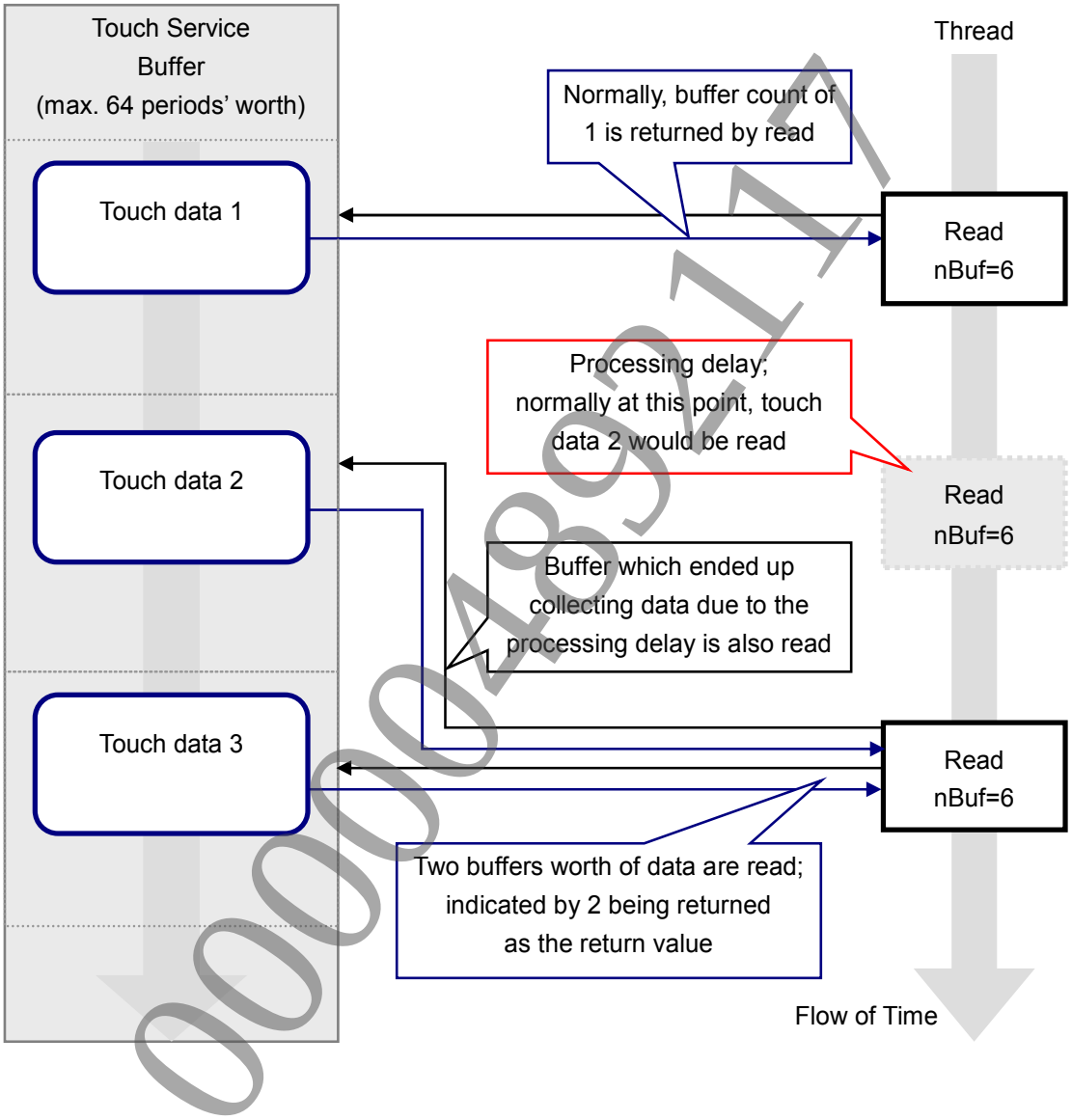
For example, if a maximum of six periods' worth of data is to be obtained, specify 6 for the buffer count to obtain. Doing this will make it possible to obtain, counting back from the newest data being buffered by the Touch Service, as many as six sets of touch data information.

The actual number of sets of data which were obtained can be ascertained based on the return value of the `sceTouchRead()` function. If this value is greater than one, this signifies that a processing delay has occurred

```
SceTouchData tdf[6], tdb[6];
int res;

// Obtain data from the front touch panel
res = sceTouchRead(SCE_TOUCH_PORT_FRONT, tdf, 6);

if(res > 1){
   //A processing delay occurred while obtaining the data of the front touch
panel.
```

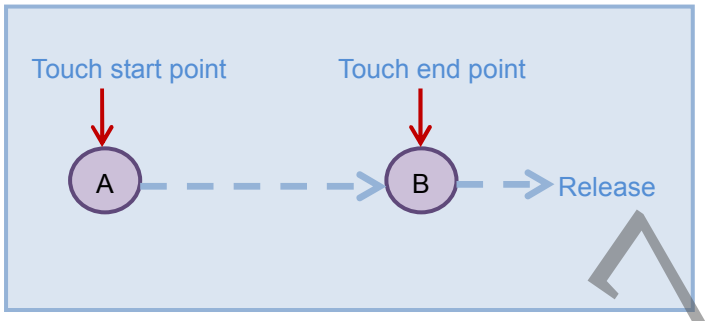
```
}

// Obtain data from the rear touch panel
res = sceTouchRead(SCE_TOUCH_PORT_BACK, tdb, 6);

if(res > 1){
    //A processing delay occurred while obtaining the data of the rear touch panel.
}
```

Within the Touch Service, up to a maximum of 64 buffers (periods' worth) of data are held. In other words, the maximum number of buffers which can be specified by the `sceTouchRead()` function is likewise 64.

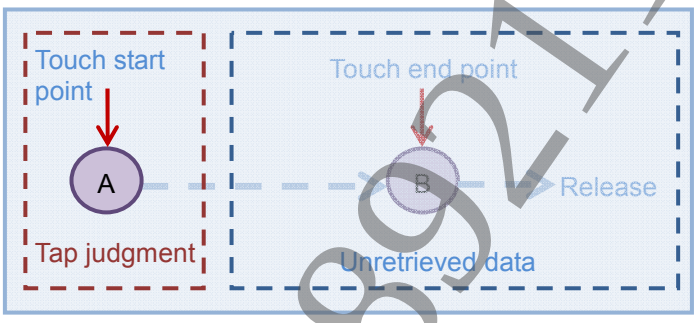


Unobtained Touch Data and Importance of History Information

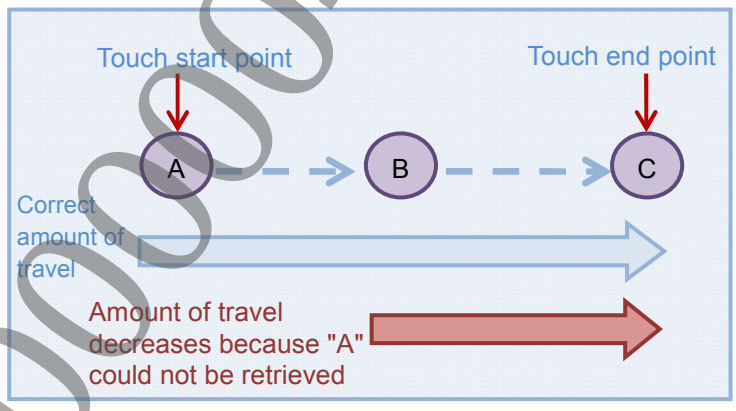
Obtaining touch data without fail is crucial to improving the stability of touch operation performed by a user. The following figure describes a movement of a touch point being dragged from the touch point "A" to "B" and then released.



In this case, if the touch point "B" is unobtained, the operation may be judged as a tap gesture at the touch point "A" because only the touch point "A" has been obtained.



The following figure shows another example. It describes a movement when a touch point travels from "A" to "C". In this case, if the touch point "A" is unobtained, the amount of travel "A" - "C" decreases to "B" - "C", and therefore, there is a possibility that some applications cannot get adequate amount of travel.



These are just a few examples of the many. For a better estimation of touch data behavior, it is important to obtain all touch data without fail. Touch data is obtained at the VSYNC period and the Touch Service holds 64 pieces of history information. An application should be implemented so as to obtain all the data by making use of the history information.

Especially, for gesture recognition using libsystemgesture, make sure to obtain all data for input to libsystemgesture, irrespective of the application's frame rate. For details on libsystemgesture, refer to the "libsystemgesture Overview" and "libsystemgesture Reference" documents.

Selecting the Touch Data Obtainment Method in Applications

Applications can obtain touch data using either the `sceTouchRead()` function or the `sceTouchPeek()` function. Since these two functions differ in their specifications as described below, the function to be used should be selected taking into consideration the characteristics, implementation, etc., of the application. For details on each of these functions, refer to "Touch Service Reference".

sceTouchRead()

`sceTouchRead()` blocks the processing from the time when the previous touch data was obtained until the time when the buffer in the Touch Service is updated. Therefore, if `sceTouchRead()` is called, obtainment of the latest data is always guaranteed. However, since the period during which the processing is blocked depends on the call timing, it is recommended to call `sceTouchRead()` from a thread that allows blocking.

sceTouchPeek()

`sceTouchPeek()` gets the data stored in the buffer in the Touch Service when this function is called. Unlike `sceTouchRead()`, there is no waiting until the buffer in the Touch Service becomes the latest data. Therefore, `sceTouchPeek()` does not block the processing and instead immediately returns the processing to the caller application. However, depending on the timing of the call, the latency from when a touch occurs may be up to 1VSYNC greater compared to `sceTouchRead()`.

Moreover, since as described above the specifications of `sceTouchPeek()` are such that there is no waiting until the buffer in the Touch Service becomes the latest data, the same data as the previous time is obtained if there is no update data. If data of the same time is not required by the application, processing that does not use unneeded data must be implemented.

3 Touch Data

Multiple Touch Points

The maximum number of touch points that can be obtained simultaneously by the Touch Service is 6 points for the front touch panel and 4 points for the rear touch panel.

Touch ID

An ID of 0 to 127 is attached to the touch data. This ID is guaranteed to remain unchanged from the time a touch panel is touched until it is released.

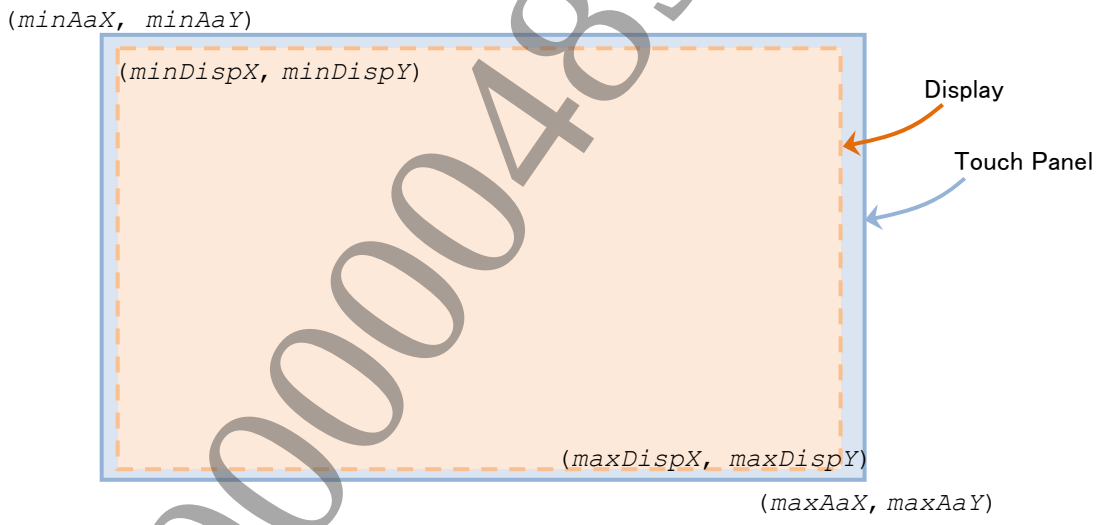
As a result, the application can know the touch point behavior by tracing the touch IDs.

A unique touch ID is allocated each time a touch panel is touched, but the IDs that are assigned are not necessarily consecutive.

Coordinate System of Touch Data

The touch panel information can be obtained in the `SceTouchPanelInfo` type by using the `sceTouchGetPanelInfo()` function.

The relationship between the active area (*`minAaX`*, *`minAaY`*, *`maxAaX`*, *`maxAay`*) of a touch panel of the `SceTouchPanelInfo` type and the display position coordinates (*`minDispX`*, *`minDispY`*, *`maxDispX`*, *`maxDispY`*) on the touch panel is generally shown as the following diagram.

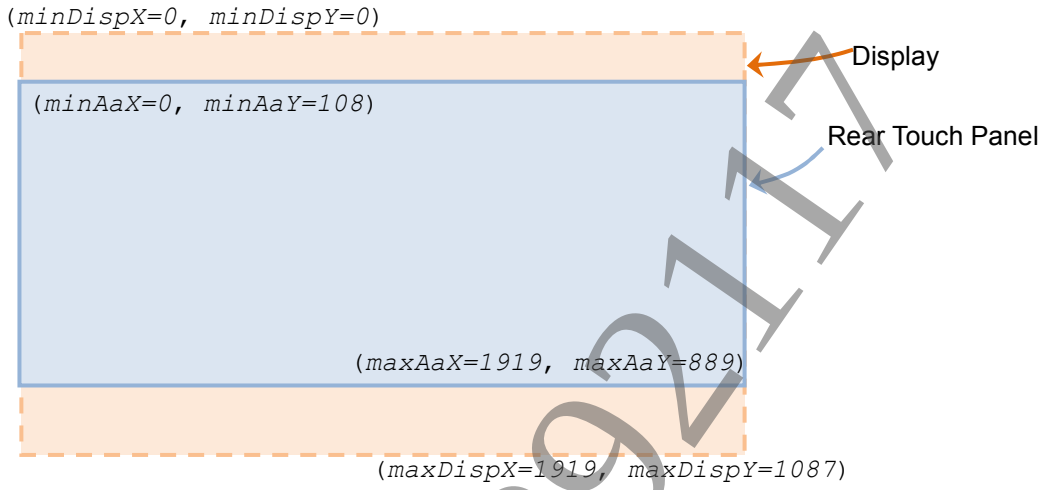


Since the display area and the touch panel's active area are the same size in the front touch panel, the front touch panel is set in the following configuration:

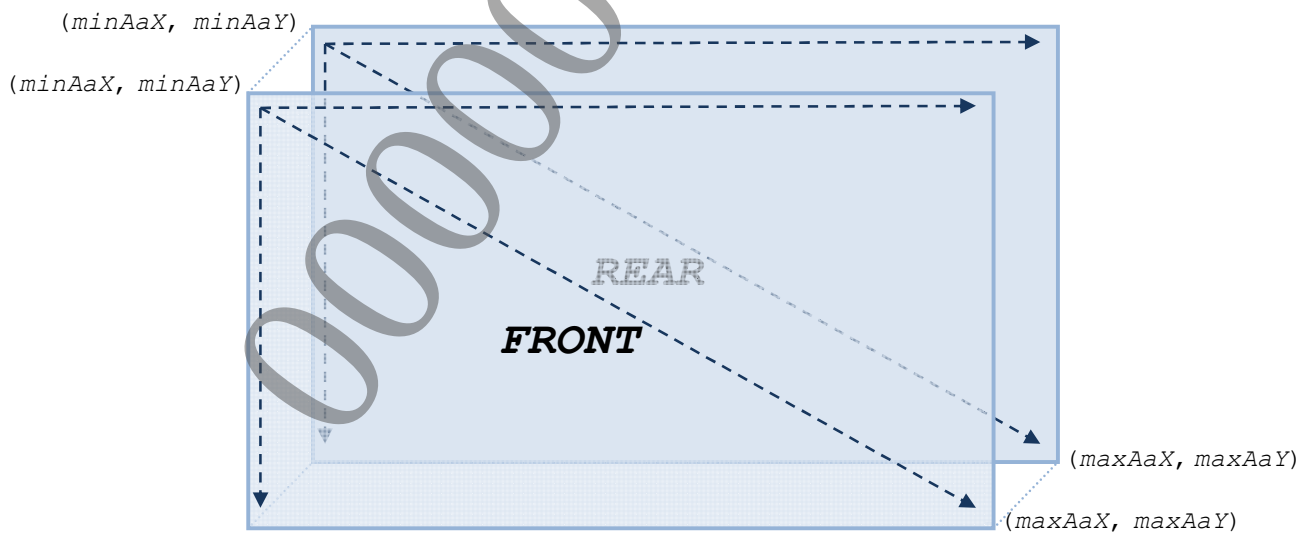
```
minAaX = minDispX = 0
minAaY = minDispY = 0
maxAaX = maxDispX = 1919
maxAaY = maxDispY = 1087
```

In addition, the Y coordinates of the rear touch panel will be small relative to the display area; therefore, configuration will be as follows:

```
minDispX = 0
maxDispX = 1919
minDispY = 0
maxDispY = 1087

minAaX = 0
maxAaX = 1919
minAaY = 108
maxAaY = 889
```



The minimum values (*minAaX*, *minAaY*) of the active area correspond to the top left of the front touch panel and the top right of the rear touch panel, and the maximum values (*maxAaX*, *maxAaY*) correspond to the bottom right of the front touch panel and the bottom left of the rear touch panel, so that when viewed from the front touch panel side, the front touch panel and the rear touch panel are positioned so as to form a paired positional relationship.



The coordinate information of the front touch panel and the coordinate information of the rear touch panel is not necessarily the same.

Obtain the coordinate information of the touch panel that will be used.

Touch information field

An *info* member variable of the `SceTouchReport` type is provided for this field, which indicates the touch information.

When the obtained data is in a specific state, this field is used to relay this information. Currently, the following states are defined.

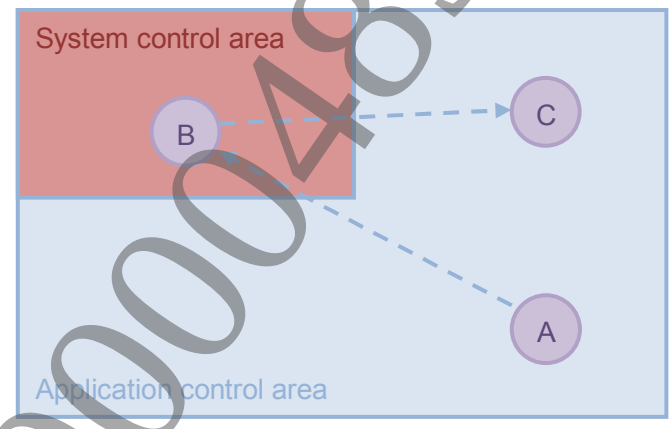
SCE_TOUCH_REPORT_INFO_HIDE_UPPER_LAYER

This is the state where the obtained touch point coordinates location falls inside an area that is controlled by a high-priority application or some services of the system. Normally, the touch data located in an area controlled by another application cannot be obtained, but if the coordinates location of the data that is first touched was in the area controlled by one's own application, touch operation can be made with `SCE_TOUCH_REPORT_INFO_HIDE_UPPER_LAYER` applied. This state is entered when, for example, a drawing is performed through the Common dialog or impose. (As an exception, this state will not be entered when a system message or "Notifications", which is shown when a game is executed in full-screen mode, is displayed because such display does not affect the touch operations.)

Performing an application operation by using a touch in this state (for example, performing select/enter using a button placed as an UI element) may cause incorrect operation. Thus use of touch data that is in this state as described above should be avoided.

This state is canceled when the control area of the higher application is exited and the area controlled by one's own application is returned to.

In the case shown in the following figure, when the data touched at point "A" in the application control area moves to "B" in the system control area, `SCE_TOUCH_REPORT_INFO_HIDE_UPPER_LAYER` occurs. Then, when the data moves to "C" in the application control area, the `SCE_TOUCH_REPORT_INFO_HIDE_UPPER_LAYER` state is canceled.



Touch Data Status

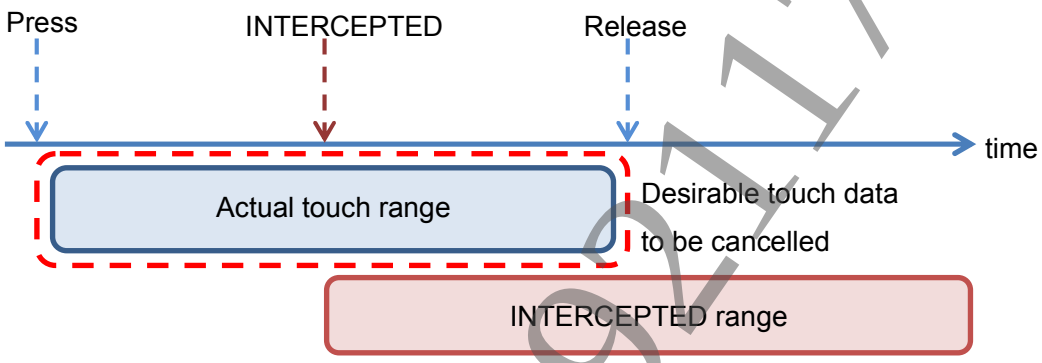
As the status information related to whole touch data, an `SceTouchData` type variable *`status`* is provided.

This field is available for all the touch points (`SceTouchReport`) obtained at that point in time.

Currently, the following statuses are defined.

SCE_TOUCH_STATUS_INTERCEPTED

This status occurs in the state where the whole touch data is intercepted by the system software. It is not possible to obtain touch data from an application if this status is established. However, if touch data has already existed, the data is valid until it is released. In this case, it is recommended to cancel the states related to all the touch data to avoid unintended select/enter operations.



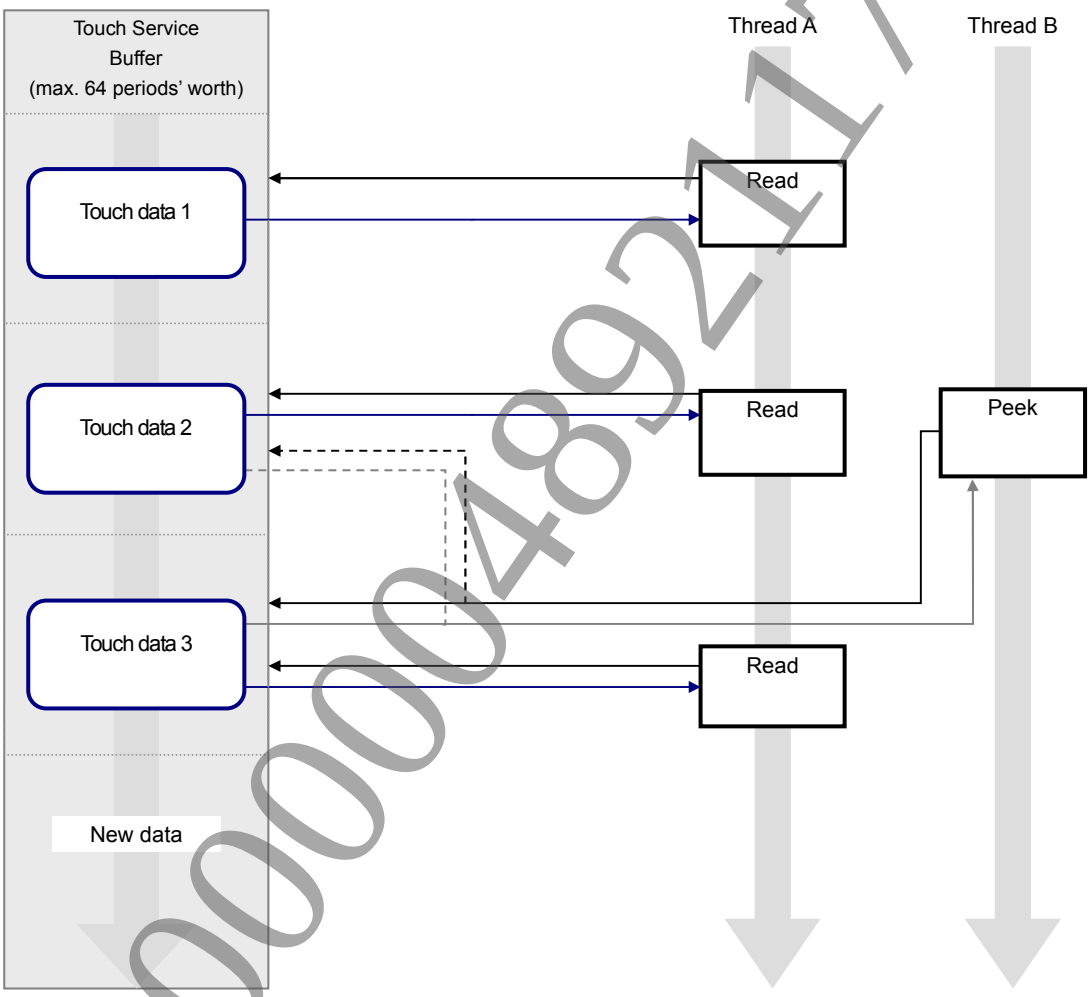
This status may be established at the same time when `SCE_TOUCH_REPORT_INFO_HIDE_UPPER_LAYER` of the touch information field is entered. In this case, the priority order for processing is determined as `SCE_TOUCH_STATUS_INTERCEPTED` > `SCE_TOUCH_REPORT_INFO_HIDE_UPPER_LAYER`.

4 Precautions

Buffer Polling

When the `sceTouchRead()` function is called, the touch data that has not yet been read since the last time the function was called is read from the buffer in the Touch Service. Since this buffer is a process global resource, inconsistencies will occur if the `sceTouchRead()` function is called from multiple threads within the same process at the same time.

Call the `sceTouchRead()` function from only one thread for that reason. If the touch data is needed by another thread, use the `sceTouchPeek()` function. By using the `sceTouchPeek()` function, it is possible to snoop the data without initializing the buffer.



However, note that since the `sceTouchPeek()` function references the contents of the buffer at the moment that the function is called, due to variations in interrupt load it may not be possible to obtain data at a consistent interval even if the touch data is being sampled every VBLANK period. This means that, in the diagram above, it is unclear whether the Peek by thread B will obtain touch data 2 or touch data 3.

Cautions Regarding Touch Point Behavior

Resolution between Two Touch Points,

When the center distance between two touch points becomes so short that it exceeds the resolution of a touch panel, it may become impossible to maintain the two points as two distinct points and they may be recognized as a single point.

In the current touch panels, the above-described phenomenon tends to occur when the center distance falls below 1.0 to 1.5 cm.

Influence of Conductive Products on Capacitive Touch Panels

A capacitive touch panel determines touch points by detecting the capacitance changes made through the human body.

Therefore, if a conductive product (such as a metal table or cabinet, or an electrical appliance) touches a touch panel, incorrect touch points may be detected owing to the capacitance changes caused by such product.

Particularly, the rear touch panel is likely to be affected by such products, and thus careful consideration must be given to the use of the rear touch panel when designing an application for which the unit is placed on a table to play.

Ownership of Touch Data

With regard to touch data, data input may be appropriated when the system software or another application operates on a priority basis.

The Touch Service judges which application's area the location that is first touched falls in, and determines the touch data ownership process accordingly. The owner of touch data the ownership process of which has been determined once does not change until release.

When not Using the Touch Panels

When not using the touch panels, in game scenes use the `sceTouchSetSamplingState()` function to turn off touch panel sampling. In this way, it is possible to achieve a slight decrease in the power consumption of the system as a whole.

Touch Force

Support for information regarding touch force ended with SDK 2.500. Be careful not to embed processing using this information in your application.

When enabling touch force information by calling `sceTouchEnableTouchForce()` in a version of SDK 2.500 or later, 0 will be set to the *`force`* member of the `SceTouchReport` structure when the touch panels are not touched and 128 will be set when touched. When touch force information is disabled by calling `sceTouchDisableTouchForce()`, 0 will always be set to the *`force`* member of the `SceTouchReport` structure regardless of touch/no touch.

5 Notes on PlayStation®TV

Behavior When the Touchscreen Pointer is Used With PlayStation®TV

With PlayStation®TV and with a Development Kit (DevKit) having **PS TV Emulation** set to **On**, the touchscreen pointer feature can be used, with emulation of the touch panels using the Controller Number 1 wireless controller instead of the touch panels themselves. When a user uses the touchscreen pointer, applications can use the Touch Service to obtain input from wireless controllers as touch operations.

Note the following when using the touchscreen pointer.

- During touchscreen pointer usage, it will not be possible to use the Controller Service to obtain input from wireless controllers.
- The maximum value for the touch points that can be obtained at the same time with touchscreen pointer is 2.
- The Touch Service API specifications do not change during touchscreen pointer usage.
- During display of the onscreen keyboard, the touchscreen pointer will be disabled. Users will operate the onscreen keyboard using a wireless controller with the touchscreen pointer feature turned off.
- When **PS TV Emulation** is turned **On** in a DevKit, the touch panels will be disabled.
- The touchscreen pointer can also be used by touching the touch pad of the DUALSHOCK®4 wireless controller. In this case, only the features of the front touch panel side can be used with the touchscreen pointer. If you want to use features of the rear touch panel side with the DUALSHOCK®4 wireless controller, use the touchscreen pointer for the back side with the R3 button in the same manner as the DUALSHOCK®3 wireless controller.

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.