PLAYSTATION TECHNICAL NOTE
===

Date: December 4, 1996
Ref:
Author: SCEI R & D

Subject: Callbacks and Critical Sections

ABSTRACT

The "Callback" mechanism is offered in the PlayStation Library in order to
handle asynchronous interrupts issued from various devices.
A function registered as a callback function will be executed
in a "critical section", a session in which interrupts are prohibited,
using a dedicated local stack.
Please refer to the "Kernel Library (libapi)" and "Controller Library (libetc)"
chapters of "Run-time Library Overview" for details.

CALLBACK PROCESSING

Some library callback functions will not be executed in a critical
section; they are processed as standard function calls.

In such a situation, there are three problems as follows:

1. When a callback function is executed in a non-critical section,
   an interrupt may occur. Thus a program that assumes no interrupts
   during execution may experience a problem.

2. When a callback function is executed in a critical section, a dedicated
   local stack is used, whereas when it is executed in a non-critical section
   the current stack is used. Thus there may be a problem if a program
   strictly controls the amount of stack used.

3. When a returning thread(*) is changed within a callback function, for
   example in "sample\etc\thread\main.c", the thread will not be changed upon
   returning from the callback function. Moreover, the program will return
   to the changed thread(*) upon returning from another callback function.

There are two callback functions that are not executed in a critical section.
Please note that these functions are executed as standard functions under
certain conditions.

1) Callback functions registered by libgpu DrawSyncCallback()

  -While waiting for drawing to terminate after calling DrawSync(0),
   a function defined by DrawSyncCallback is processed as a standard function
   since it guarantees that DrawSync(0) waits till drawing termination.

  -The drawing functions such as LoadImage(), StoreImage(), ClearImage(),
   DrawOTag(), etc. are executed as standard functions only when the amount
   of the drawing data is very small. When the drawing data is small,

the drawing will be finished during context switching to a critical section. In order to avoid the CPU time being wasted like this, the function is called as a standard function when the data is under approximately 16 x 16 pixels.

2) Callback functions registered by libsnd SsSetMarkCallback()

  -SsSetTickMode() with Tick mode = SS_NOTICK, and SsSeqCalledTbyT() are called in foreground within an event loop. They are executed as standard functions.

All other functions are executed in a critical section.

For DrawSync() and libsnd, it is possible to know whether a function is executed in a critical section or not at coding phase. However, other callback functions that are invoked by the drawing instructions, whether they are being executed in a critical section (callback context) or not have to be checked at actual execution time by using the function CheckCallback().

This function is valid only within "Callback".

Please note that the event handlers are not effective since all of them are executed in an interrupt prohibited section / local stack.

*** End of document ***