==
T H E " U N - O F F I C I A L"
PLAYSTATION DEVELOPMENT FAQ

SIO
CONFERENCE
==

Release v1.2
Last Updated: 2/29/96

DISCLAIMER

This FAQ is to aid in informing the licensed game developer about the development environment provided by Sony Computer Entertainment.

The Development System Tool to which this manual relates is supplied pursuant to and subject to the terms of the Sony Playstation Licensed Developer Agreement.

This FAQ is intended for distribution to and use only by Sony PlayStation Licensed Developers in accordance with the Sony Playstation Licensed Developer Agreement. The information in this manual is subject to change without notice.

The content of this manual is Confidential Information of Sony for the purposes of the Sony PlayStation Licensed Developer Agreement and otherwise.

TRADEMARK INFORMATION

PlayStation and Sony Computer Entertainment names and logos are trade names and/or trademarks and/or copyright artwork of Sony Corporation(or its subsidiaries).

All specific names included herein are trademarks and are so acknowledged: IBM, Microsoft, MS-DOS. Any trademarks not mentioned here are still hypothetically acknowledged.

COPYRIGHT NOTICE

[1.] INPUT/OUTPUT (Peripherals)

[1.1.]: Controllers

[1.1.1.]: PadInit is documented to return a long, giving the number of control pads connected, but it always returns 1. Why?

PadInit is actually a void function, though it is not declared that way. Ignore any returned value, as it is not valid. Use the InitPAD, StartPAD, and ChangeClearPAD(0) combination for nonstandard controllers, and to get more information about connection status.

[1.1.2.]: PadRead seems to return the data for both pad 1 and pad 2 with a single call, is this correct?

PadRead does return the data for both pads. The first two bytes of the long are pad 1's data, while pad 2's data is in the upper two bytes.

[1.2.]: Link Cable?

[1.2.1.]: Should I use ioctl, or _comb_control for link cable programming?

Use _comb_control. It replaces ioctl for use with the sio: device, and gives much greater functionality.

[1.3.]: Memory Card?

[1.3.1.] What is a "new card" mentioned in the memory card event class, EvSpNEW?

The "new card" indicates "no writing/reading after connection". When EvSpNEW is returned by the _card_info() execution, it means the card is connected.

[1.3.2.]: How can the insertion and extraction of the card be detected?

The extraction can be detected by executing the _card_info() function by polling, and by checking if the event changes from EvSpIOE to EvSpTIMOUT. The insertion can be detected by checking whether EvSpNEW is returned in the _card_info() execution or not. Refer to Library Reference Vol. 1(DTL-D2140A) for details. Note that the insertion and extraction may cause the access failure to the memory card or the inverse failure because the card and the controller are connected onto the same serial line.

[1.3.3.]: Why does the time-out take place at the timing when the event of card_info() is informed if the controller inserted into the same port as the memory card is extracted?

It is proper operation because the card and the controller are connected onto the same serial line. Note that the insertion and extraction of the

controller may cause the access failure to the memory card or the inverse failure.

[1.3.4.]: The initialization order of PadInit(or InitPAD), and InitCARD, is important.

When InitCARD is called with an argument of '1'', it enables access to controller and memory card. For this to work correctly/consistently, the pad must be initialized FIRST! Initialize in the order shown below.

```
InitPAD(...);
   :
InitCARD(1);
```