NAME

ddi_dma_alloc_handle – allocate DMA handle

SYNOPSIS

#include <sys/ddi.h>
#include <sys/sunddi.h>

int ddi_dma_alloc_handle(dev_info_t **dip*, **ddi_dma_attr_t** **attr*,
    int (**callback***)(caddr_t), caddr_t** *arg*, **ddi_dma_handle_t** **handlep***);**

ARGUMENTS

dip Pointer to the device's **dev_info** structure.

attr Pointer to a DMA attribute structure for this device (see **ddi_dma_attr**(9S)).

callback The address of a function to call back later if resources aren't available now. The following special function addresses may also be used.

DDI_DMA_SLEEP wait until resources are available

DDI_DMA_DONTWAIT do not wait until resources are available and do not schedule a callback.

arg Argument to be passed to a callback function, if such a function is specified.

handlep Pointer to the DMA handle to be initialized.

INTERFACE LEVEL

Solaris DDI specific (Solaris DDI).

DESCRIPTION

ddi_dma_alloc_handle( ) allocates a new DMA handle. A DMA handle is an opaque object used as a reference to subsequently allocated DMA resources. **ddi_dma_alloc_handle( )** accepts as parameters the device information referred to by *dip* and the device's DMA attributes described by a **ddi_dma_attr**(9S) structure. A successful call to **ddi_dma_alloc_handle( )** fills in the value pointed to by *handlep*. A DMA handle must only be used by the device for which it was allocated and is only valid for one I/O transaction at a time.

The callback function, *callback*, indicates how a caller wants to handle the possibility of resources not being available. If *callback* is set to **DDI_DMA_DONTWAIT**, then the caller does not care if the allocation fails, and can handle an allocation failure appropriately. If *callback* is set to **DDI_DMA_SLEEP**, then the caller wishes to have the the allocation routines wait for resources to become available. If any other value is set, and a DMA resource allocation fails, this value is assumed to be a function to call at a later time when resources may become available. When the specified function is called, it is passed *arg* as an argument. The specified callback function must return either **DDI_DMA_CALLBACK_RUNOUT** or **DDI_DMA_CALLBACK_DONE**. **DDI_DMA_CALLBACK_RUNOUT** indicates that the callback routine attempted to allocate DMA resources but failed to do so, in which case the callback function is put back on a list to be called again later. **DDI_DMA_CALLBACK_DONE** indicates either success at allocating DMA resources or the driver no longer wishes to retry.

The callback function is called in interrupt context. Therefore, only system functions that are accessible from interrupt context is available. The callback function must take whatever steps necessary to protect its critical resources, data structures, queues, so forth.

When a DMA handle is no longer needed, **ddi_dma_free_handle**(9F) must be called to free the handle.

RETURN VALUES

ddi_dma_alloc_handle( ) returns:

DDI_SUCCESS Successfully allocated a new DMA handle.

DDI_DMA_BADATTR
The attributes specified in the **ddi_dma_attr**(9S) structure make it impossible for the system to allocate potential DMA resources.

DDI_DMA_NORESOURCES
No resources are available.

CONTEXT

ddi_dma_alloc_handle( ) can be called from user, kernel, or interrupt context, except when *callback* is set to **DDI_DMA_SLEEP**, in which case it can be called from user or kernel context only.

SEE ALSO

ddi_dma_addr_bind_handle(9F), **ddi_dma_buf_bind_handle**(9F), **ddi_dma_burstsizes**(9F), **ddi_dma_free_handle**(9F), **ddi_dma_unbind_handle**(9F), **ddi_dma_attr**(9S)

Writing Device Drivers