NAME

tran_start – request to transport a SCSI command

SYNOPSIS

#include <sys/scsi/scsi.h>

int *prefix* **tran_start(struct scsi_address** **ap*, **struct scsi_pkt** **pkt***);**

INTERFACE LEVEL

Solaris architecture specific (Solaris DDI).

ARGUMENTS

pkt Pointer to the **scsi_pkt**(9S) structure that is about to be transferred.

ap Pointer to a **scsi_address**(9S) structure.

DESCRIPTION

The **tran_start( )** vector in the **scsi_hba_tran**(9S) structure must be initialized during the HBA driver's **attach**(9E) to point to an HBA entry point to be called when a target driver calls **scsi_transport**(9F).

tran_start( ) must perform the necessary operations on the HBA hardware to transport the SCSI command in the *pkt* structure to the target/logical unit device specified in the *pkt* structure.

If the flag **FLAG_NOINTR** is set in **pkt_flags** in *pkt*, **tran_start( )** should not return until the command has been completed. The command completion callback **pkt_comp** in *pkt* must not be called for commands with **FLAG_NOINTR** set, since the return is made directly to the function invoking **scsi_transport**(9F).

When the flag **FLAG_NOINTR** is not set, **tran_start( )** must queue the command for execution on the hardware and return immediately. The member **pkt_comp** in *pkt* indicates a callback routine to be called upon command completion.

Refer to **scsi_pkt**(9S) for other bits in **pkt_flags** for which the HBA driver may need to adjust how the command is managed.

If the **auto_rqsense** capability has been set, and the status length allocated in **tran_init_pkt**(9E) is greater than or equal to **sizeof(struct scsi_arq_status)**, automatic request sense is enabled for this *pkt*. If the command terminates with a Check Condition, the HBA driver must arrange for a Request Sense command to be transported to that target/logical unit, and the members of the **scsi_arq_status** structure pointed to by **pkt_scbp** updated with the results of this Request Sense command before the HBA driver completes the command pointed by *pkt*.

The member **pkt_time** in *pkt* is the maximum number of seconds in which the command should complete. A **pkt_time** of zero means no timeout should be performed.

For a command which has timed out, the HBA driver must perform some recovery operation to clear the command in the target, typically an Abort message, or a Device or Bus Reset. The **pkt_reason** member of the timed-out *pkt* should be set to **CMD_TIMEOUT**, and **pkt_statistics** OR'ed with **STAT_TIMEOUT**. If the HBA driver can successfully recover from the timeout, **pkt_statistics** must also be OR'ed with one of **STAT_ABORTED**, **STAT_BUS_RESET** or **STAT_DEV_RESET**, as appropriate. This informs the target driver that timeout recovery has already been successfully accomplished for the timed-out command. The **pkt_comp** completion callback, if not **NULL**, must also be called at the conclusion of the timeout recovery.

If the timeout recovery was accomplished with an Abort Tag message, only the timed-out packet is affected, and the packet must be returned with **pkt_statistics** OR'ed with **STAT_ABORTED** and **STAT_TIMEOUT**.

If the timeout recovery was accomplished with an Abort message, all commands active in that target are affected. All corresponding packets must be returned with **pkt_reason**, **CMD_TIMEOUT**, and **pkt_statistics** OR'ed with **STAT_TIMEOUT** and **STAT_ABORTED**.

If the timeout recovery was accomplished with a Device Reset, all packets corresponding to commands active in the target must be returned in the transport layer for this target. Packets corresponding to commands active in the target must be returned returned with **pkt_reason** set to **CMD_TIMEOUT**, and **pkt_statistics** OR'ed with **STAT_DEV_RESET** and **STAT_TIMEOUT**. Currently inactive packets queued for the device should be returned with **pkt_reason** set to **CMD_RESET** and **pkt_statistics** OR'ed with **STAT_ABORTED**.

If the timeout recovery was accomplished with a Bus Reset, all packets corresponding to commands active in the target must be returned in the transport layer. Packets corresponding to commands active in the target must be returned with **pkt_reason** set to **CMD_TIMEOUT** and **pkt_statistics** OR'ed with **STAT_TIMEOUT** and **STAT_BUS_RESET**. All queued packets for other targets on this bus must be returned with **pkt_reason** set to **CMD_RESET** and **pkt_statistics** OR'ed with **STAT_ABORTED**.

Note that, after either a Device Reset or a Bus Reset, the HBA driver must enforce a reset delay time of **'scsi-reset-delay'** milliseconds, during which time no commands should be sent to that device, or any device on the bus, respectively.

tran_start( ) should initialize the following members in *pkt* to **0**. Upon command completion, the HBA driver should ensure that the values in these members are updated to accurately reflect the states through which the command transitioned while in the transport layer.

pkt_resid For commands with data transfer, this member must be updated to indicate the residual of the data transferred.

pkt_reason The reason for the command completion. This field should be set to **CMD_CMPLT** at the beginning of **tran_start( )**, then updated if the command ever transitions to an abnormal termination state. To avoid losing information, do not set **pkt_reason** to any other error state unless it still has its original **CMD_CMPLT** value.

pkt_statistics Bit field of transport-related statistics

pkt_state Bit field with the major states through which a SCSI command can transition.

Note: the members listed above, and **pkt_hba_private** member, are the only fields in the **scsi_pkt**(9S) structure which may be modified by the transport layer.

RETURN VALUES

tran_start( ) must return:

TRAN_ACCEPT	The packet was accepted by the transport layer.
TRAN_BUSY	The packet could not be accepted because there was already a packet in progress for this target/logical unit, the HBA queue was full, or the target device queue was full.
TRAN_BADPKT	The DMA count in the packet exceeded the DMA engine's maximum DMA size, or the packet could not be accepted for other reasons.
TRAN_FATAL_ERROR	A fatal error has occurred in the HBA.

SEE ALSO

attach(9E), **tran_init_pkt**(9E), **scsi_hba_attach**(9F), **scsi_transport**(9F), **scsi_address**(9S), **scsi_hba_tran**(9S), **scsi_pkt**(9S)

Writing Device Drivers