Table Of Contents

Document information

History

V0.1 - Initial draft.

Disclaimer

The information contained in this document is preliminary design information for an Amiga implementation of USB. All information herein is subject to change without prior notice.

Use of the information in this document is at own risk. In no way will the authors or distributors of this document be liable for any losses directly or indirectly related to information used from this document.

USB Host Controller Driver API

The following sections describe a proposal for the API between the USB System Software and the USB Host Controller Drivers (HCDs).
USB Host Controller drivers are Exec devices, using an extension of the standard IoRequest for data passing, and library calls for specific functions.
It is not the intension of this document to specify the inner workings of the USB System Software, although some hints are given thru function arguments and the occasional implementation note.

Data Flow

The following functions are used for data passing between the USB System Software (which may be passing it on from a USB Function driver) and Functions on the USB bus. The functions are in fact messages sendt to the Host Controller driver using device requests. The messages are described as functions here to keep functionality descriptions standardized at this stage of designing.

USBHCSend

LONG USBHCSend(hcendpoint, data, size)

This function is used by the System to request sending of a block of data to an EndPoint of a USB Function. The data block given to this function is not guaranteed to be of the EndPoint's specified max. data packet size. This function must internally be able to split transfers into multiple packets if necessary.
If the EndPoint has transfer size restrictions, the HCD must be able to generate errors accordingly, if the size restrictions are not obeyed (e.g. last packet does not fill buffer, etc.)
The function must return an error code indicating succes/the reason of failure. Zero indicates success.
This call is always asyncronous. Syncronous operation is performed by the caller by using waiting for the request to return (IoRequest).

hcendpoint	-	HCD-generated reference to the EndPoint to send data to. This reference is the value returned by USBHCAddEndPoint() for the EndPoint in question.
data	-	Pointer to the data to send
size	-	Bytesize of the data buffer.

USBHCReceive

LONG USBHCReceive(hcendpoint, data, size)

This function is called by the System to requests data retrieval from an EndPoint of a USB Function. Up to <size> bytes of data must be read into the specified data buffer from the EndPoint.
This function must transparently break up the request into smaller USB packets to

meet the max. packet size of the EndPoint. Any constraints on packet size for the EndPoint must still be obeyed. If such requirements are present, and are not aboyed, this function must return an error code indicating so.
The function must return an error code indicating succes/the reason of failure. Zero indicates success.
This call is always asyncronous. Syncronous operation is performed by the caller by using waiting for the request to return (IoRequest).

hcendpoint - HCD-generated reference to the EndPoint to send data to. This reference is the value returned by USBHCAddEndPoint() for the EndPoint in question.
data - Pointer to the data to send
size - Bytesize of the data buffer.

USB Function control

The following functions are used for maintaining USB Functions between the USB System Software and the HCD. The functions are placed in the HCD.

USBHCAddFunction

hcfunction = USBHCAddFunction(hub, usfunction, taglist)

This function is called by the USB System when a new function insertion has been detected by a hub driver. As such, this call is normally a result of a call to USBAddFunction() in the USB System Function driver API.
The HCD is expected to setup its internal structures needed for handling the new function (not its endpoints!) and returning a reference to it.
All information about the new USB Function is supplied using a tag list. This abstracts the USB Function information from the HCD, allowing easier extensions to the USB Function information in the future.

The <hub> argument is the reference returned by a USBHCAddFunction() call for the hub device into which the new device has been inserted. This allows the HCD to keep its topology information up to date.
In one occasion the <hub> argument will be NULL. That is when, at USB System startup, the System wants a reference to the Host Controller's root hub. Whenever this function is called with NULL as <hub> argument, a reference to the root hub must be returned, allowing the USB System to hook into the USB topology of the Host Controller.

The <usfunction> argument holds a USB System Software private reference to the USb Function being added. This reference must be used by the HCD if refering to the Function in a call to the USB System. This is currently only used for the root hub driver, when calling USBAddFunction() to add a USB Function being inserted at the root hub. In the future this reference may be needed for other things as well.

hub - HCD-generated reference to the USB hub Function into which

the new Function has been inserted.

usfunction - A USB System Software private reference to the Function being added.

taglist - Pointer to a taglist holding information on the USB Function just inserted.

USBHCRemFunction

void USBHCRemFunction(hcfunction)

This function is called when a USB Function has been removed from the USB topology of the Host Controller. Such a call will normally be the result of a USB hub driver calling the USB System USBRemFunction() function.
When this call is made the HCD must abort any outstanding requests to the EndPoints of the USB Function, and should release ressources related to the Function.

hcfunction - HCD-generated reference to the USB Function which has been removed from the USB bus.

USB EndPoint control

The following functions are used for maintaining USB Functions between the USB System Software and the HCD.

USBHCAddEndPoint

hcendpoint = USBHCAddEndPoint(hcfunction, usendpoint, taglist)

With this function the USB System will add EndPoints to a USB Function already added to the HCD. In this way the System is responsible for setting up communication endpoints for USB Functions, separating the HCD from USB Function configuration selection.
The HCD must determine if the USB bus it covers can sustain the data transfer requested by the EndPoint (as described in the USB spec. about accepting EndPoints). If not, the HCD muse refuse the EndPoint by returning NULL. Otherwise this function must return a HCD-private reference, which will be used by the System for future references to the EndPoint.

hcfunction - HCD-generated reference to the USB Function which has been removed from the USB bus.

usendpoint - A USB System Software private reference to the EndPoint being added. Use this for USB System calls referencing the EndPoint.

taglist - Pointer to a taglist holding information about the EndPoint.

USBHCRemEndPoint

void USBHCRemEndPoint(hcendpoint)

This function is called by the System to remove an EndPoint previously inserted by a call to USBHCAddEndPoint().
The HCD must abort all requests pending for the EndPoint in question, and should release any ressources allocated for the EndPoint.

This function will typically be called for all EndPoints of a USB Function if a Function driver chooses to change the USB Function's configuration.

hcendpoint - HCD-generated reference to the EndPoint which must be removed.