NAME

iicmux - I2C bus mulitiplexer framework

SYNOPSIS

To compile this driver into the kernel, place the following line in your kernel configuration file:

device iicmux

Alternatively, to load the driver as a module at boot time, place the following line in loader.conf(5):

iicmux_load="YES"

Note that it is usually not necessary to explicitly load the driver module, as it will be loaded automatically along with the driver for the specific mux hardware in use.

DESCRIPTION

The **iicmux** framework provides support code to help implement drivers for various I2C bus multiplexer (mux) hardware. **iicmux** is not a standalone driver, it is a collection of support functions and driver methods which are used by individual mux hardware drivers. It will be loaded automatically when needed by a mux hardware driver. This manual page provides an overview of the I2C mux framework and its behavior.

Generally speaking, an I2C mux is connected to an upstream I2C bus, and to one or more downstream I2C buses, and it can be commanded to connect any one of the downstream buses to the upstream bus. Some hardware may be able to connect multiple downstream buses at the same time, but that concept is not supported by **iicmux**.

The **iicmux** framework operates automatically when I2C slave devices initiate I/O. It does not require (or even allow for) any external control to select the active downstream bus.

When there is no I/O in progress, the mux is said to be in the "idle" state. Some mux hardware has the ability to disconnect all downstream buses when in an idle state. Other hardware must always have one of the downstream buses connected. Individual mux hardware drivers typically provide a way to select which downstream bus (if any) should be connected while in the idle state. In the absence of such configuration, whichever downstream bus was last used remains connected to the upstream bus.

When an I2C slave device on a bus downstream of a mux initiates I/O, it first requests exclusive use of the bus by calling **iicbus_request_bus**(). This request is communicated to the bus’s parent, which is the **iicmux** framework mux driver. Once exclusive bus ownership is obtained, the mux driver connects the upstream I2C bus to the downstream bus which hosts the slave device that requested bus ownership.

The mux hardware maintains that upstream-to-downstream connection until the slave device calls **iicbus_release_bus**(). Before releasing ownership, the mux driver returns the mux hardware to the idle state.

FDT CONFIGURATION

On an fdt(4) based system, an I2C mux device node is defined as a child node of its upstream I2C bus when the mux device is an I2C slave itself. It may be defined as a child node of any other bus or device in the system when it is not an I2C slave, in which case the *i2c-parent* property indicates which upstream bus the mux is attached to. In either case, the children of the mux node are additional I2C buses, which will have one or more I2C slave devices described in their child nodes.

Drivers using the **iicmux** framework conform to the standard i2c/i2c-mux.txt bindings document.

HINTS CONFIGURATION

On a device.hints(5) based system, these values are configurable for **iicmux** framework drivers :

hint.<driver>.<unit>.at

The upstream iicbus(4) the **iicmux** instance is attached to.

When configured via hints, the driver automatically adds an iicbus instance for every downstream bus supported by the chip. There is currently no way to indicate used versus unused downstream buses.

SEE ALSO

iicbus(4)

HISTORY

The **iicmux** framework first appeared in FreeBSD 13.0.