Linux kernel SLIMbus support

Overview

What is SLIMbus?

SLIMbus (Serial Low Power Interchip Media Bus) is a specification developed by MIPI (Mobile Industry Processor Interface) alliance. The bus uses master/slave configuration, and is a 2-wire multi-drop implementation (clock, and data).

Currently, SLIMbus is used to interface between application processors of SoCs (System-on-Chip) and peripheral components (typically codec). SLIMbus uses Time-Division-Multiplexing to accommodate multiple data channels, and a control channel.

The control channel is used for various control functions such as bus management, configuration and status updates. These messages can be unicast (e.g. reading/writing device specific values), or multicast (e.g. data channel reconfiguration sequence is a broadcast message announced to all devices)

A data channel is used for data-transfer between 2 SLIMbus devices. Data channel uses dedicated ports on the device.

Hardware description:

SLIMbus specification has different types of device classifications based on their capabilities. A manager device is responsible for enumeration, configuration, and dynamic channel allocation. Every bus has 1 active manager.

A generic device is a device providing application functionality (e.g. codec).

Framer device is responsible for clocking the bus, and transmitting frame-sync and framing information on the bus.

Each SLIMbus component has an interface device for monitoring physical layer.

Typically each SoC contains SLIMbus component having 1 manager, 1 framer device, 1 generic device (for data channel support), and 1 interface device. External peripheral SLIMbus component usually has 1 generic device (for functionality/data channel support), and an associated interface device. The generic device's registers are mapped as 'value elements' so that they can be written/read using SLIMbus control channel exchanging control/status type of information. In case there are multiple framer devices on the same bus, manager device is responsible to select the active-framer for clocking the bus.

Per specification, SLIMbus uses "clock gears" to do power management based on current frequency and bandwidth requirements. There are 10 clock gears and each gear changes the SLIMbus frequency to be twice its previous gear.

Each device has a 6-byte enumeration-address and the manager assigns every device with a 1-byte logical address after the devices report presence on the bus.

Software description:

There are 2 types of SLIMbus drivers:

slim_controller represents a 'controller' for SLIMbus. This driver should implement duties needed by the SoC (manager device, associated interface device for monitoring the layers and reporting errors, default framer device).

slim_device represents the 'generic device/component' for SLIMbus, and a slim_driver should implement driver for that slim_device.

Device notifications to the driver:

Since SLIMbus devices have mechanisms for reporting their presence, the framework allows drivers to bind when corresponding devices report their presence on the bus. However, it is possible that the driver needs to be probed first so that it can enable corresponding SLIMbus device (e.g. power it up and/or take it out of reset). To support that behavior, the framework allows drivers to probe first as well (e.g. using standard DeviceTree compatibility field). This creates the necessity for the driver to know when the device is functional (i.e. reported present). device_up callback is used for that reason when the device reports present and is assigned a logical address by the controller.

Similarly, SLIMbus devices 'report absent' when they go down. A 'device_down' callback notifies the driver when the device reports absent and its logical address assignment is invalidated by the controller.

Another notification "boot_device" is used to notify the slim_driver when controller resets the bus. This notification allows the driver to take necessary steps to boot the device so that it's functional after the bus has been reset.

Driver and Controller APIs:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\(linux-master)(Documentation)(driver-api)slimbus.rst, line 94)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: include/linux/slimbus.h
   :internal:
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\(linux-master)(Documentation)(driver-api)slimbus.rst, line 97)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/slimbus/slimbus.h
   :internal:
```

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\(linux-master)(Documentation)(driver-api)slimbus.rst, line 100)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/slimbus/core.c
   :export:
```

Clock-pause:

SLIMbus mandates that a reconfiguration sequence (known as clock-pause) be broadcast to all active devices on the bus before the bus can enter low-power mode. Controller uses this sequence when it decides to enter low-power mode so that corresponding clocks and/or power-rails can be turned off to save power. Clock-pause is exited by waking up framer device (if controller driver initiates exiting low power mode), or by toggling the data line (if a slave device wants to initiate it).

Clock-pause APIs:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\(linux-master)(Documentation)(driver-api)slimbus.rst, line 115)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/slimbus/sched.c
   :export:
```

Messaging:

The framework supports regmap and read/write apis to exchange control-information with a SLIMbus device. APIs can be synchronous or asynchronous. The header file <linux/slimbus.h> has more documentation about messaging APIs.

Messaging APIs:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\(linux-master)(Documentation)(driver-api)slimbus.rst, line 126)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/slimbus/messaging.c
   :export:
```

Streaming APIs:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\(linux-master)(Documentation)(driver-api)slimbus.rst, line 131)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: drivers/slimbus/stream.c
   :export:
```