NAME

spigen - SPI generic I/O device driver

SYNOPSIS

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

```
device spi
device spibus
device spigen
```

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

```
spigen_load="YES"
```

DESCRIPTION

The **spigen** driver provides direct access to a slave device on the SPI bus. Each instance of a **spigen** device is associated with a single chip-select line on the bus, and all I/O performed through that instance is done with that chip-select line asserted.

SPI data transfers are inherently bi-directional; there are no separate read and write operations. When commands and data are sent to a device, data also comes back from the device, although in some cases the data may not be useful (or even documented or predictable for some devices). Likewise on a read operation, whatever data is in the buffer at the start of the operation is sent to (and typically ignored by) the device, with each outgoing byte then replaced in the buffer by the corresponding incoming byte. Thus, all buffers passed to the transfer functions are both input and output buffers.

The **spigen** driver provides access to the SPI slave device with the following ioctl(2) calls, defined in <*sys/spigenio.h*>:

SPIGENIOC_TRANSFER (*struct spigen_transfer*)

Transfer a command and optional associated data to/from the device, using the buffers described by the st_command and st_data fields in the *spigen_transfer*. Set *st_data.iov_len* to zero if there is no data associated with the command.

```
struct spigen_transfer {
        struct iovec st_command;
        struct iovec st_data;
};
```

SPIGENIOC_TRANSFER_MMAPPED (*spigen_transfer_mmapped*)

Transfer a command and optional associated data to/from the device. The buffers for the transfer are a previously-mmap'd region. The length of the command and data within that region are described by the *stm_command_length* and *stm_data_length* fields of *spigen_transfer_mmapped*. If *stm_data_length* is non-zero, the data appears in the memory region immediately following the command (that is, at offset *stm_command_length* from the start of the mapped region).

```
struct spigen_transfer_mmapped {
        size_t stm_command_length;
        size_t stm_data_length;
};
```

SPIGENIOC_GET_CLOCK_SPEED (*uint32_t*)
: Get the maximum clock speed (bus frequency in Hertz) to be used when communicating with this slave device.

SPIGENIOC_SET_CLOCK_SPEED (*uint32_t*)
: Set the maximum clock speed (bus frequency in Hertz) to be used when communicating with this slave device. The setting remains in effect for subsequent transfers; it is not necessary to reset this before each transfer. The actual bus frequency may be lower due to hardware limitations of the SPI bus controller device.

SPIGENIOC_GET_SPI_MODE (*uint32_t*)
: Get the SPI mode (clock polarity and phase) to be used when communicating with this device.

SPIGENIOC_SET_SPI_MODE (*uint32_t*)
: Set the SPI mode (clock polarity and phase) to be used when communicating with this device. The setting remains in effect for subsequent transfers; it is not necessary to reset this before each transfer.

HINTS CONFIGURATION

On a device.hints(5) based system, such as MIPS, these values are configurable for **spigen**:

hint.spigen.%d.at
: The spibus the **spigen** instance is attached to.

hint.spigen.%d.clock
: The maximum bus frequency to use when communicating with this device. Actual bus speed may be lower, depending on the capabilities of the SPI bus controller hardware.

hint.spigen.%d.cs

The chip-select number to assert when performing I/O for this device. Set the high bit (1 << 31) to invert the logic level of the chip select line.

hint.spigen.%d.mode
The SPI mode (0-3) to use when communicating with this device.

FDT CONFIGURATION

On an fdt(4) based system, the spigen device is defined as a slave device subnode of the SPI bus controller node. All properties documented in the *spibus.txt* bindings document can be used with the **spigen** device. The most commonly-used ones are documented below.

The following properties are required in the **spigen** device subnode:

compatible
Must be the string "freebsd,spigen".

reg Chip select address of device.

spi-max-frequency
The maximum bus frequency to use when communicating with this slave device. Actual bus speed may be lower, depending on the capabilities of the SPI bus controller hardware.

The following properties are optional for the **spigen** device subnode:

spi-cpha
Empty property indicating the slave device requires shifted clock phase (CPHA) mode.

spi-cpol
Empty property indicating the slave device requires inverse clock polarity (CPOL) mode.

spi-cs-high
Empty property indicating the slave device requires chip select active high.

FILES

*/dev/spigen**

SEE ALSO

fdt(4), device.hints(5), spi(8)

HISTORY

The **spigen** driver appeared in FreeBSD 11.0. FDT support appeared in FreeBSD 11.2.