Hardware drivers for the Raspberry Pi

Duss Pirmin

May 27, 2015

1 Summary

The Broadcom BCM2835/2836 used in the Raspberry Pi provides the following periperal interfaces:

- a mini UART (no RTS/CTS lines)
- two SPI masters
- two I2C
- 54 GPIO lines

Depending on the revision of the Raspberry Pi board this functions are mapped to a 26 or 40 pin connector. Non GPIO functions are implemented as alternate functions on the GPIO pins.

Links to some useful web sites with information about the Raspberry Pi:

- `https://www.raspberrypi.org/documentation/hardware/raspberrypi/bcm2835` The document `BCM2835-ARM-Periperals.pdf` describes the functions of the different phe-ripheral systems and the Registers
- `http://raspi.tv/2014/rpi-gpio-quick-reference-updated-for-raspberry-pi-b` Describes the pinout of the different Raspberry Pi models
- `http://elinux.org/RPi\_Low-level\_peripherals` Shows some example code how to program the registers

2 Software structure

Figure 1 shows the drivers that are used to control the sub systems. The driver rpi-iocon controls the configuration of the registers that define the function of the pins. All other drivers request the required pin function there.

2.1 rpi-iocon

This process controls the registers that define what function are on the different pins of the connetor. It is possible to configure some or all pin functions int the <config> node.

This sevice could maybe placed in the platform_drv.

2.1.1 required services

- MMIO

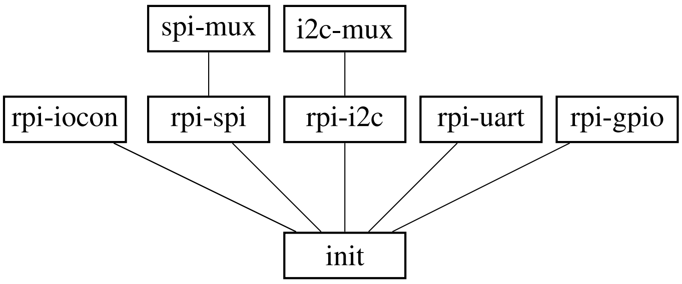

Figure 1: drivers used to control the subsystems of the Raspberry Pi

2.1.2 provided service

- IoConfig

2.1.3 Configuration

- board revision
- fix pins

2.2 rpi-spi

Controls the communication over one of the SPI ports. The constructor throws an exception Hardware_not_avaliable when it isn't able to configure the required SPI port via the IcCon service.

2.2.1 required services

- MMIO
- IoConfig

2.2.2 provided service

- SPI

2.2.3 Configuration

- port number (0/1)
- communication speed (1..5)MHz (TODO check possible values here)

2.3 spi-mux

This is the resource multiplexer for the SPI protocl. Multiple programs can use a single SPI port by using this multiplexer.

2.3.1 required services

- SPI

2.3.2 provided service

- SPI

2.3.3 Configuration

None at the moment. The port is selected throug the service routing in the configuration of the core.

2.4 rpi-i2c

Controls the communication over one of the I2c ports. The constructor throws an exception Hardware_not_avaliable when it isn't able to configure the required I2C port via the IcCon service.

2.4.1 required services

- MMIO
- IoConfig

2.4.2 provided service

- I2C

2.4.3 Configuration

- port number (0/1)
- communication speed (100..400)kHz (TODO check possible values here)

2.5 i2c-mux

This is the resource multiplexer for the I2C protocl. Multiple programs can use a single I2C port by using this multiplexer.

2.5.1 required services

- I2C

2.5.2 provided service

- SPI

2.5.3 Configuration

None at the moment. The port is selected throug the service routing in the configuration of the core.

2.6 rpi-i2c

This program already exists, but needs modification to use the IoCon service.

2.7 rpi-gpio

Controls the GPIO. A process that wants to use a GPIO needs to request the gpio first. This request throws an Hardware_not_avaliable when the port is already in use.

2.8 rpi-uart

This is the already existing driver for the UART port of the Raspberry Pi. It needs some modifications to use the IoConfig service to configure the required pins.

2.8.1 required services

- MMIO
- IoConfig

2.8.2 provided service

- UART