Driver for I2C radios for the Silicon Labs Si4713 FM Radio Transmitters

Contact: Eduardo Valentin <eduardo.valentin@nokia.com>

Information about the Device

This chip is a Silicon Labs product. It is a I2C device, currently on 0x63 address.
Basically, it has transmission and signal noise level measurement features.

The Si4713 integrates transmit functions for FM broadcast stereo transmission.
The chip also allows integrated receive power scanning to identify low signal power FM channels.

The chip is programmed using commands and responses. There are also several properties which can change the behavior of this chip.

Users must comply with local regulations on radio frequency (RF) transmission.

Device driver description

There are two modules to handle this device. One is a I2C device driver and the other is a platform driver.

The I2C device driver exports a v4l2-subdev interface to the kernel.
All properties can also be accessed by v4l2 extended controls interface, by using the v4l2-subdev calls (g_ext_ctrls, s_ext_ctrls).

The platform device driver exports a v4l2 radio device interface to user land.
So, it uses the I2C device driver as a sub device in order to send the user commands to the actual device. Basically it is a wrapper to the I2C device driver.

Applications can use v4l2 radio API to specify frequency of operation, mute state,
etc. But mostly of its properties will be present in the extended controls.

When the v4l2 mute property is set to 1 (true), the driver will turn the chip off.

Properties description

The properties can be accessed using v4l2 extended controls.
Here is an output from v4l2-ctl util:

```
/ # v4l2-ctl -d /dev/radio0 --all -L
Driver Info:
        Driver name   : radio-si4713
        Card type     : Silicon Labs Si4713 Modulator
        Bus info      :
        Driver version: 0
        Capabilities  : 0x00080800
                RDS Output
                Modulator
```

```
Audio output: 0 (FM Modulator Audio Out)
Frequency: 1408000 (88.000000 MHz)
Video Standard = 0x00000000
Modulator:
        Name                 : FM Modulator
        Capabilities         : 62.5 Hz stereo rds
        Frequency range      : 76.0 MHz - 108.0 MHz
        Subchannel modulation: stereo+rds

User Controls

                           mute (bool) : default=1 value=0

FM Radio Modulator Controls

           rds_signal_deviation (int)  : min=0 max=90000 step=10 default=200
value=200 flags=slider
                 rds_program_id (int)  : min=0 max=65535 step=1 default=0
value=0
               rds_program_type (int)  : min=0 max=31 step=1 default=0 value=0
                    rds_ps_name (str)  : min=0 max=96 step=8 value='si4713  '
                 rds_radio_text (str)  : min=0 max=384 step=32 value=''
  audio_limiter_feature_enabled (bool) : default=1 value=1
     audio_limiter_release_time (int)  : min=250 max=102390 step=50 default=5010
value=5010 flags=slider
        audio_limiter_deviation (int)  : min=0 max=90000 step=10 default=66250
value=66250 flags=slider
audio_compression_feature_enabl (bool) : default=1 value=1
         audio_compression_gain (int)  : min=0 max=20 step=1 default=15 value=15
flags=slider
    audio_compression_threshold (int)  : min=-40 max=0 step=1 default=-40
value=-40 flags=slider
  audio_compression_attack_time (int)  : min=0 max=5000 step=500 default=0
value=0 flags=slider
 audio_compression_release_time (int)  : min=100000 max=1000000 step=100000
default=1000000 value=1000000 flags=slider
     pilot_tone_feature_enabled (bool) : default=1 value=1
           pilot_tone_deviation (int)  : min=0 max=90000 step=10 default=6750
value=6750 flags=slider
           pilot_tone_frequency (int)  : min=0 max=19000 step=1 default=19000
value=19000 flags=slider
          pre_emphasis_settings (menu) : min=0 max=2 default=1 value=1
               tune_power_level (int)  : min=0 max=120 step=1 default=88
value=88 flags=slider
          tune_antenna_capacitor (int)  : min=0 max=191 step=1 default=0
value=110 flags=slider
/ #
```

Here is a summary of them:

* Pilot is an audible tone sent by the device.

pilot_frequency - Configures the frequency of the stereo pilot tone.
pilot_deviation - Configures pilot tone frequency deviation level.
pilot_enabled - Enables or disables the pilot tone feature.

* The si4713 device is capable of applying audio compression to the transmitted signal.

acomp_enabled - Enables or disables the audio dynamic range control feature.
acomp_gain - Sets the gain for audio dynamic range control.
acomp_threshold - Sets the threshold level for audio dynamic range control.
acomp_attack_time - Sets the attack time for audio dynamic range control.
acomp_release_time - Sets the release time for audio dynamic range control.

* Limiter setups audio deviation limiter feature. Once a over deviation occurs, it is possible to adjust the front-end gain of the audio input and always prevent over deviation.

limiter_enabled - Enables or disables the limiter feature.
limiter_deviation - Configures audio frequency deviation level.
limiter_release_time - Sets the limiter release time.

* Tuning power

power_level - Sets the output power level for signal transmission.
antenna_capacitor - This selects the value of antenna tuning capacitor manually or automatically if set to zero.

* RDS related

rds_ps_name - Sets the RDS ps name field for transmission.
rds_radio_text - Sets the RDS radio text for transmission.
rds_pi - Sets the RDS PI field for transmission.
rds_pty - Sets the RDS PTY field for transmission.

* Region related

preemphasis - sets the preemphasis to be applied for transmission.

RNL

This device also has an interface to measure received noise level. To do that, you should
ioctl the device node. Here is an code of example:

```
int main (int argc, char *argv[])
{
        struct si4713_rnl rnl;
        int fd = open("/dev/radio0", O_RDWR);
        int rval;

        if (argc < 2)
                return -EINVAL;

        if (fd < 0)
                return fd;

        sscanf(argv[1], "%d", &rnl.frequency);

        rval = ioctl(fd, SI4713_IOC_MEASURE_RNL, &rnl);
        if (rval < 0)
                return rval;

        printf("received noise level: %d\n", rnl.rnl);

        close(fd);
}
```

The struct si4713_rnl and SI4713_IOC_MEASURE_RNL are defined under
include/media/si4713.h.

Stereo/Mono and RDS subchannels

The device can also be configured using the available sub channels for
transmission. To do that use S/G_MODULATOR ioctl and configure txsubchans
properly.
Refer to the V4L2 API specification for proper use of this ioctl.

Testing

Testing is usually done with v4l2-ctl utility for managing FM tuner cards.
The tool can be found in v4l-dvb repository under v4l2-apps/util directory.

Example for setting rds ps name:

```
# v4l2-ctl -d /dev/radio0 --set-ctrl=rds_ps_name="Dummy"
```