```c
/* Copyright (c) 2011, RidgeRun
 * All rights reserved.
 *
From https://www.ridgerun.com/developer/wiki/index.php/Gpio-int-test.c

 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 3. All advertising materials mentioning features or use of this software
 *    must display the following acknowledgement:
 *    This product includes software developed by the RidgeRun.
 * 4. Neither the name of the RidgeRun nor the
 *    names of its contributors may be used to endorse or promote products
 *    derived from this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY RIDGERUN ''AS IS'' AND ANY
 * EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
 * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
 * DISCLAIMED. IN NO EVENT SHALL RIDGERUN BE LIABLE FOR ANY
 * DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
 * (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
 * LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
 * ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
 * SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

#include "gpio-utils.h"
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <string.h>

/****************************************************************
 * gpio_export
 ****************************************************************/
int gpio_export(unsigned int gpio)
{
    int fd, len;
    char buf[MAX_BUF];

    fd = open(SYSFS_GPIO_DIR "/export", O_WRONLY);
    if (fd < 0) {
        perror("gpio/export");
        return fd;
    }

    len = snprintf(buf, sizeof(buf), "%d", gpio);
```

```
55      write(fd, buf, len);
56      close(fd);
57
58      return 0;
59  }
60
61  /****************************************************************
62   * gpio_unexport
63   ****************************************************************/
64  int gpio_unexport(unsigned int gpio)
65  {
66      int fd, len;
67      char buf[MAX_BUF];
68
69      fd = open(SYSFS_GPIO_DIR "/unexport", O_WRONLY);
70      if (fd < 0) {
71          perror("gpio/export");
72          return fd;
73      }
74
75      len = snprintf(buf, sizeof(buf), "%d", gpio);
76      write(fd, buf, len);
77      close(fd);
78      return 0;
79  }
80
81  /****************************************************************
82   * gpio_set_dir
83   ****************************************************************/
84  int gpio_set_dir(unsigned int gpio, const char* dir)
85  {
86      int fd, len;
87      char buf[MAX_BUF];
88
89      len = snprintf(buf, sizeof(buf), SYSFS_GPIO_DIR  "/gpio%d/direction", gpio);
90
91      fd = open(buf, O_WRONLY);
92      if (fd < 0) {
93          perror("gpio/direction");
94          return fd;
95      }
96
97      write(fd, dir, sizeof(dir)+1);
98
99      close(fd);
100     return 0;
101 }
102
103 /****************************************************************
104  * gpio_set_value
105  ****************************************************************/
106 int gpio_set_value(unsigned int gpio, unsigned int value)
107 {
108     int fd, len;
```

```
109         char buf[MAX_BUF];
110
111         len = snprintf(buf, sizeof(buf), SYSFS_GPIO_DIR "/gpio%d/value", gpio);
112
113         fd = open(buf, O_WRONLY);
114         if (fd < 0) {
115             perror("gpio/set-value");
116             return fd;
117         }
118
119         if (value)
120             write(fd, "1", 2);
121         else
122             write(fd, "0", 2);
123
124         close(fd);
125         return 0;
126     }
127
128     /****************************************************************
129      * gpio_get_value
130      ****************************************************************/
131     int gpio_get_value(unsigned int gpio, unsigned int *value)
132     {
133         int fd, len;
134         char buf[MAX_BUF];
135         char ch;
136
137         len = snprintf(buf, sizeof(buf), SYSFS_GPIO_DIR "/gpio%d/value", gpio);
138
139         fd = open(buf, O_RDONLY);
140         if (fd < 0) {
141             perror("gpio/get-value");
142             return fd;
143         }
144
145         read(fd, &ch, 1);
146
147         if (ch != '0') {
148             *value = 1;
149         } else {
150             *value = 0;
151         }
152
153         close(fd);
154         return 0;
155     }
156
157
158     /****************************************************************
159      * gpio_set_edge
160      ****************************************************************/
161
162     int gpio_set_edge(unsigned int gpio, const char *edge)
```

```
163 {
164     int fd, len;
165     char buf[MAX_BUF];
166 
167     len = snprintf(buf, sizeof(buf), SYSFS_GPIO_DIR "/gpio%d/edge", gpio);
168 
169     fd = open(buf, O_WRONLY);
170     if (fd < 0) {
171         perror("gpio/set-edge");
172         return fd;
173     }
174 
175     write(fd, edge, strlen(edge) + 1);
176     close(fd);
177     return 0;
178 }
179 
180 /****************************************************************
181  * gpio_fd_open
182  ****************************************************************/
183 
184 int gpio_fd_open(unsigned int gpio, unsigned int dir)
185 {
186     int fd, len;
187     char buf[MAX_BUF];
188 
189     len = snprintf(buf, sizeof(buf), SYSFS_GPIO_DIR "/gpio%d/value", gpio);
190 
191     fd = open(buf, dir | O_NONBLOCK );
192     if (fd < 0) {
193         perror("gpio/fd_open");
194     }
195     return fd;
196 }
197 
198 /****************************************************************
199  * gpio_fd_close
200  ****************************************************************/
201 
202 int gpio_fd_close(int fd)
203 {
204     return close(fd);
205 }
206 
207 
```