

Switch_toggle



Sanjay T

Library file



RuggedBoard A5D2X GPIO Switch-Toggle



Objective

Toggle three LEDs (GPIO 77, 81, 83) using a user push-button switch connected to GPIO 76.



Pin Configuration

Signal	Description	GPIO No.	Path
LED1	PC13	77	<code>/sys/class/gpio/gpio77</code>
LED2	PC17	81	<code>/sys/class/gpio/gpio81</code>
LED3	PC19	83	<code>/sys/class/gpio/gpio83</code>
Switch	Push-button/PC12	76	<code>/sys/class/gpio/gpio76</code>



Folder Structure

```
rbgpio_led_toggle/  
├── include/  
│   └── rbgpio.h  
├── src/  
│   └── rbgpio.c  
├── main.c  
└── Makefile
```



`include/rbgpio.h`

```

#ifndef RB_GPIO_H
#define RB_GPIO_H

int gpio_export(int pin);
int gpio_unexport(int pin);
int gpio_set_direction(int pin, const char *direction);
int gpio_write(int pin, int value);
int gpio_read(int pin);

#endif

```



src/rbgpio.c

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <fcntl.h>
#include "rbgpio.h"

#define SYSFS_GPIO_DIR "/sys/class/gpio"

int gpio_export(int pin) {
    char buffer[64];
    int fd = open(SYSFS_GPIO_DIR "/export", O_WRONLY);
    if (fd < 0) return -1;

    snprintf(buffer, sizeof(buffer), "%d", pin);
    write(fd, buffer, strlen(buffer));
    close(fd);
    return 0;
}

```

```

int gpio_unexport(int pin) {
    char buffer[64];
    int fd = open(SYSFS_GPIO_DIR "/unexport", O_WRONLY);
    if (fd < 0) return -1;

    snprintf(buffer, sizeof(buffer), "%d", pin);
    write(fd, buffer, strlen(buffer));
    close(fd);
    return 0;
}

int gpio_set_direction(int pin, const char *direction) {
    char path[64];
    snprintf(path, sizeof(path), SYSFS_GPIO_DIR "/gpio%d/direction", pin);
    int fd = open(path, O_WRONLY);
    if (fd < 0) return -1;

    write(fd, direction, strlen(direction));
    close(fd);
    return 0;
}

int gpio_write(int pin, int value) {
    char path[64];
    snprintf(path, sizeof(path), SYSFS_GPIO_DIR "/gpio%d/value", pin);
    int fd = open(path, O_WRONLY);
    if (fd < 0) return -1;

    write(fd, value ? "1" : "0", 1);
    close(fd);
    return 0;
}

int gpio_read(int pin) {
    char path[64], value_str[3];

```

```

snprintf(path, sizeof(path), SYSFS_GPIO_DIR "/gpio%d/value", pin);
int fd = open(path, O_RDONLY);
if (fd < 0) return -1;

read(fd, value_str, sizeof(value_str));
close(fd);
return atoi(value_str);
}

```

main.c — Switch Toggle Logic

```

#include <unistd.h>
#include <stdio.h>
#include "rbgpio.h"

int leds[] = {77, 81, 83};
int switch_pin = 76;

int main() {
    gpio_export(switch_pin);
    gpio_set_direction(switch_pin, "in");

    for (int i = 0; i < 3; i++) {
        gpio_export(leds[i]);
        gpio_set_direction(leds[i], "out");
        gpio_write(leds[i], 0);
    }

    int prev = gpio_read(switch_pin);

    while (1) {
        int curr = gpio_read(switch_pin);

```

```

    if (curr != prev && curr == 1) {
        static int toggle = 0;
        toggle = !toggle;

        for (int i = 0; i < 3; i++) {
            gpio_write(leds[i], toggle);
        }

        printf("LEDs toggled %s\n", toggle ? "ON" : "OFF");
    }

    prev = curr;
    usleep(200000);
}

return 0;
}

```



Makefile

```

CC ?= ${CC}
CFLAGS = -Wall -linclude
LIB_SRC = src/rbgpio.c
MAIN = main.c

all: switch_toggle

switch_toggle:
    $(CC) $(CFLAGS) $(LIB_SRC) $(MAIN) -o switch_toggle

clean:
    rm -f switch_toggle

```

Build & Run

```
./opt/poky-tiny/2.5.2/environment-setup-cortexa5hf-neon-poky-linux-musleabi
```

```
make clean  
make
```

Transfer & execute on RuggedBoard:


```
sudo cp switch_toggle /srv/tftp  
//then  
//in rugged board  
tftp -r switch_toggle -g 192.168.1.15 //replace ur ipaddress of host machine
```


Then


```
chmod +x switch_toggle  
./switch_toggle
```

output be like:

```
root@rugged-board-a5d2x-sd1:~# ./switch_toggle  
🔄 Toggled LEDs to ON //when we press the user switch,It change for toogle the  
🔄 Toggled LEDs to OFF  
🔄 Toggled LEDs to ON  
🔄 Toggled LEDs to OFF  
🔄 Toggled LEDs to ON
```

 Toggled LEDs to OFF

 Toggled LEDs to ON

 Toggled LEDs to OFF