SPI

How to use SPI interface using the Expansion Board.

Reading the SPI Flash device ID on Expansion-Board

To use spidev on your **ODROID-XU3**, first fix the device tree source.

To use the SPI, you first need to update the kernel version to “3.10.82” or higher.

```
$ sudo apt-get update && sudo apt-get dist-upgrade
```

SPI MODE 3 does not work for unknown reason. Please try to use the SPI MODE 0.

How to enable driver (Only Ubuntu 16.04 or higher is required)

Edit /etc/modprobe.d/blacklist-odroid.conf to comment out following two lines.

```
blacklist spidev
blacklist spi_s3c64xx
```

Reboot.

```
$ reboot
```

Check your SPI node.

```
$ ls /dev/spidev*
```

Compile & run SPI test example source code

```
$ gcc -o spidev_test spidev_test.c
```

```
odroid@odroid:~$ ./spidev_test
spi mode:
bits per word: 8
max speed: 500000 Hz (500 KHz)
```

```
34 34 34 34
```

If you tie the MOSI and MISO pins, the output is “AB 00 00 00”.
We call it loopback test.

spidev_test.c

```
/*
 * SPI testing utility (using spidev driver)
 *
 * Copyright (c) 2007  MontaVista Software, Inc.
 * Copyright (c) 2007  Anton Vorontsov <avorontsov@ru.mvista.com>
 *
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 2 of the License.
 *
 * Cross-compile with cross-gcc -I/path/to/cross-kernel/include
 */

#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <sys/ioctl.h>
#include <linux/spi/spidev.h>

#define ARRAY_SIZE(a) (sizeof(a) / sizeof((a)[0]))

static void pabort(const char *s)
{
        perror(s);
        abort();
}

static const char *device = "/dev/spidev1.0";
static uint8_t mode;
static uint8_t bits = 8;
static uint32_t speed = 500000;

static void transfer(int fd)
{
        int ret;
        uint8_t tx[] = {
                0xAB, 0x00, 0x00, 0x00
        };
        uint8_t rx[ARRAY_SIZE(tx)] = {, };

        struct spi_ioc_transfer tr[2];
        tr[].tx_buf = (unsigned long)tx;
```

```
        tr[].rx_buf = (unsigned long)rx;
        tr[].len = ARRAY_SIZE(tx);
        tr[].speed_hz = speed;
        tr[].bits_per_word = bits;
        tr[].delay_usecs = ;
        tr[].cs_change = ;

        tr[1].tx_buf = (unsigned long)tx;
        tr[1].rx_buf = (unsigned long)rx;
        tr[1].len = ARRAY_SIZE(tx);
        tr[1].speed_hz = speed;
        tr[1].bits_per_word = bits;
        tr[1].delay_usecs = ;
        tr[1].cs_change = ;

        ret = ioctl(fd, SPI_IOC_MESSAGE(2), tr);
        if (ret < 1)
                pabort("can't send spi message");

        for (ret = ; ret < ARRAY_SIZE(tx); ret++) {
                if (!(ret % 6))
                        puts("");
                printf("%.2X ", rx[ret]);
        }
        puts("");
}

int main(void)
{
        int ret = ;
        int fd;

        fd = open(device, O_RDWR);
        if (fd < )
                pabort("can't open device");

        /*
         * spi mode
         */
        ret = ioctl(fd, SPI_IOC_WR_MODE, &mode);
        if (ret == -1)
                pabort("can't set spi mode");

        ret = ioctl(fd, SPI_IOC_RD_MODE, &mode);
        if (ret == -1)
                pabort("can't get spi mode");

        /*
         * bits per word
         */
        ret = ioctl(fd, SPI_IOC_WR_BITS_PER_WORD, &bits);
```

```
        if (ret == -1)
                pabort("can't set bits per word");

        ret = ioctl(fd, SPI_IOC_RD_BITS_PER_WORD, &bits);
        if (ret == -1)
                pabort("can't get bits per word");

        /*
         * max speed hz
         */
        ret = ioctl(fd, SPI_IOC_WR_MAX_SPEED_HZ, &speed);
        if (ret == -1)
                pabort("can't set max speed hz");

        ret = ioctl(fd, SPI_IOC_RD_MAX_SPEED_HZ, &speed);
        if (ret == -1)
                pabort("can't get max speed hz");

        printf("spi mode: %d\n", mode);
        printf("bits per word: %d\n", bits);
        printf("max speed: %d Hz (%d KHz)\n", speed, speed/1000);

        transfer(fd);

        close(fd);

        return ret;
}
```

If you want to use 16bit/32bit length protocol with different SPI speed on XU4, refer this c1_hardware_spidev.