GM8136

# General-Purpose Input/Output (GPIO)

User Guide
Rev.: 1.0
Issue Date: September 2014

# REVISION HISTORY

## GM8136 GPIO User Guide

| Date | Rev. | From | To |
|---|---|---|---|
| Sept. 2014 | 1.0 | - | Original |

# TABLE OF CONTENTS

# LIST OF FIGURES

# Chapter 1

# GPIO Configuration

This chapter contains the following sections:

## 1.1 GPIO Overview

General-Purpose Input/Output (a.k.a. GPIO) is a common interface of SoC. Users can configure a GPIO pin as an input or output. When the GPIO pin is configured as an output, this pin can be set to the high voltage or low voltage. When the GPIO pin is configured as an input, this pin can be used to detect the voltage level. Moreover, an input pin may be configured as an external interrupt source.

GM8136 provides two ports for GPIO. Each port has 32 pins, which can be individually configured. Please make sure that the pins are well multiplexed before using them. Figure 1-1 illustrates the control flow when a kernel GPIO function is called by the driver.

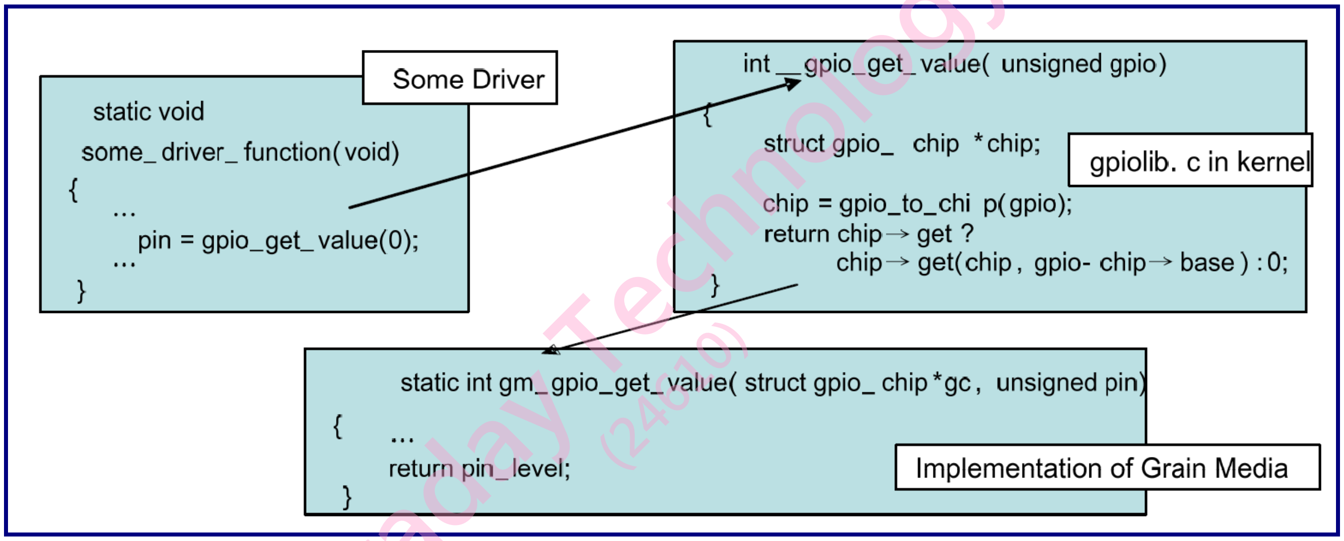

Some Driver
static void
some_ driver_ function( void)
{
...
pin = gpio_get_ value(0);
...
}
int __gpio_get_ value( unsigned gpio)
{
struct gpio_ chip *chip;
gpiolib. c in kernel
chip = gpio_to_chi p(gpio);
return chip→ get ?
chip→ get(chip, gpio- chip→ base) : 0;
}
static int gm_gpio_get_value( struct gpio_ chip *gc, unsigned pin)
{
...
return pin_level;
}
Implementation of Grain Media

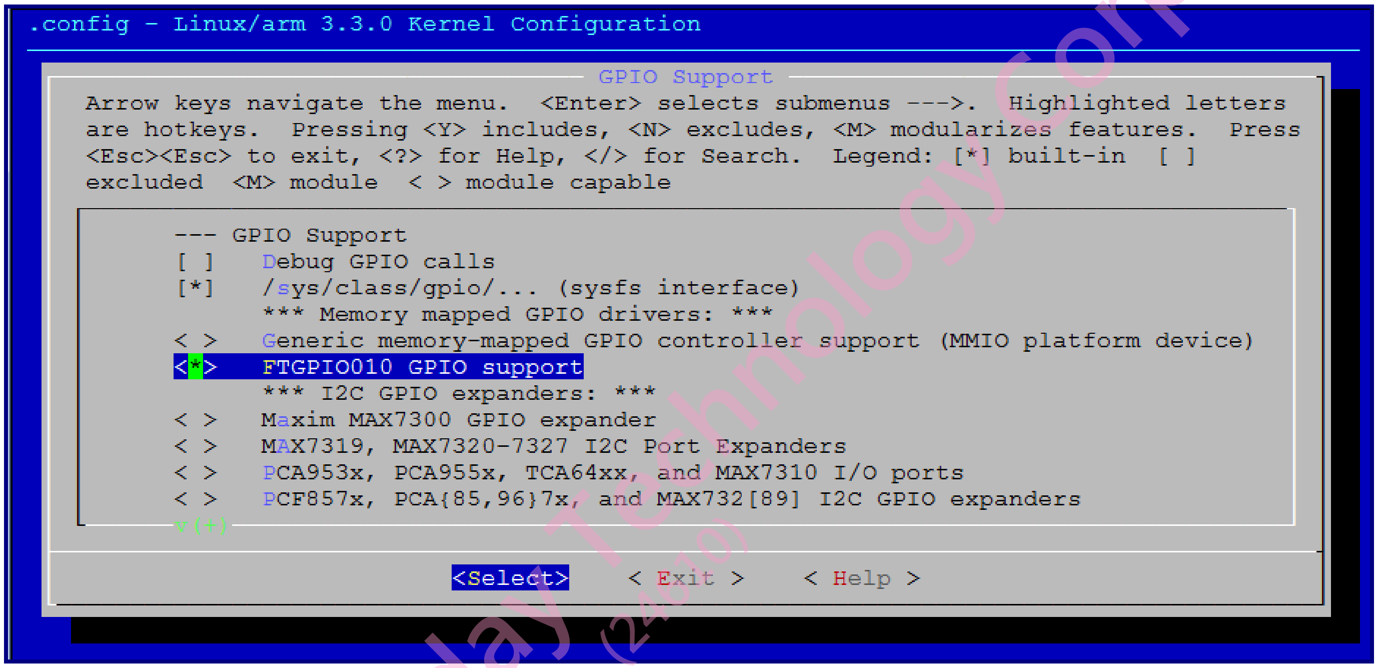
**Figure 1-1. Control Flow of GPIO**

When the driver calls one of the functions, gpio_get_value, of the kernel GPIO, kernel will call the corresponding GPIO chips based on the GPIO pin number. GM8136 has two GPIO ports. The first port controls pins[31:0] and the second port controls pins[63:32]. Users should use pins[95:0] to configure the pin and the corresponding port. Specific GPIO chip will call a callback function that has registered in advance. This implementation is a specific GPIO implementation of GM8136.

## 1.2 Options about GPIO in Linux Kernel

The GPIO function on SoC is FTGPIO010. The GPIO module is built-in in the Linux kernel image. Users can use the menu configuration (make menuconfig) to choose the items for FTGPIO010. As shown in Figure 1-2, the items are chosen in menu configuration of Linux kernel.

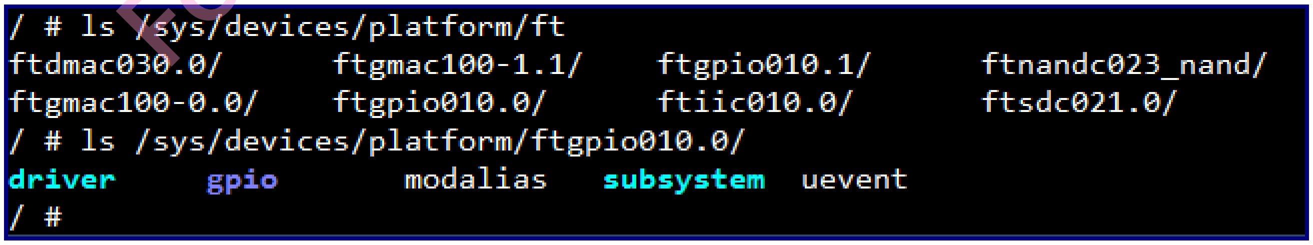
The steps are: Device Drivers ---> GPIO Support ---> FTGPIO010 GPIO support



```
.config - Linux/arm 3.3.0 Kernel Configuration
                          GPIO Support
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters
  are hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes features.  Press
  <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ]
  excluded  <M> module  < > module capable

          --- GPIO Support
          [ ]   Debug GPIO calls
          [*]   /sys/class/gpio/... (sysfs interface)
                *** Memory mapped GPIO drivers: ***
          < >   Generic memory-mapped GPIO controller support (MMIO platform device)
          <*>   FTGPIO010 GPIO support
                *** I2C GPIO expanders: ***
          < >   Maxim MAX7300 GPIO expander
          < >   MAX7319, MAX7320-7327 I2C Port Expanders
          < >   PCA953x, PCA955x, TCA64xx, and MAX7310 I/O ports
          < >   PCF857x, PCA{85,96}7x, and MAX732[89] I2C GPIO expanders
          v(+)

                     <Select>    < Exit >    < Help >
```

Figure 1-2. Build FTGPIO010

After building GPIO module in the kernel image, users can boot Linux and find sysfs of FTGPIO010, whose path is "/sys/devices/platform/ftgpio010.0". In Figure 1-3, users can find sysfs and make sure that the driver is already built in.



```
/ # ls /sys/devices/platform/ft
ftdmac030.0/      ftgmac100-1.1/    ftgpio010.1/      ftnandc023_nand/
ftgmac100-0.0/    ftgpio010.0/      ftiic010.0/       ftsdc021.0/
/ # ls /sys/devices/platform/ftgpio010.0/
driver     gpio       modalias   subsystem  uevent
/ #
```

Figure 1-3. sysfs of FTGPIO010

## 1.3 Examples of GPIO Usage

The usage of the Linux GPIO function is descried in Section 1.3. If users want to set GPIO port 0 pin0 of GM8136 as an output with high voltage, please check the following example code:

```
/* the procedure of setting gpio 0 high */
int ret = -1;
if ((ret = gpio_request(0, "gpio0")) != 0) { return ret; }
if ((ret = gpio_direction_output(0, 1)) != 0) { return ret; }
```

The input pin of GPIO should be used as an interrupt source to prevent the kernel from checking the pin status by using a periodic timer. The Linux GPIO middleware does not provide the GPIO interrupt interface; however, Grain Media has added these functions for the user convenience. Figure 1-4 is a snip example of using the GPIO input as an interrupt code, in which pin0 is set as an interrupt.

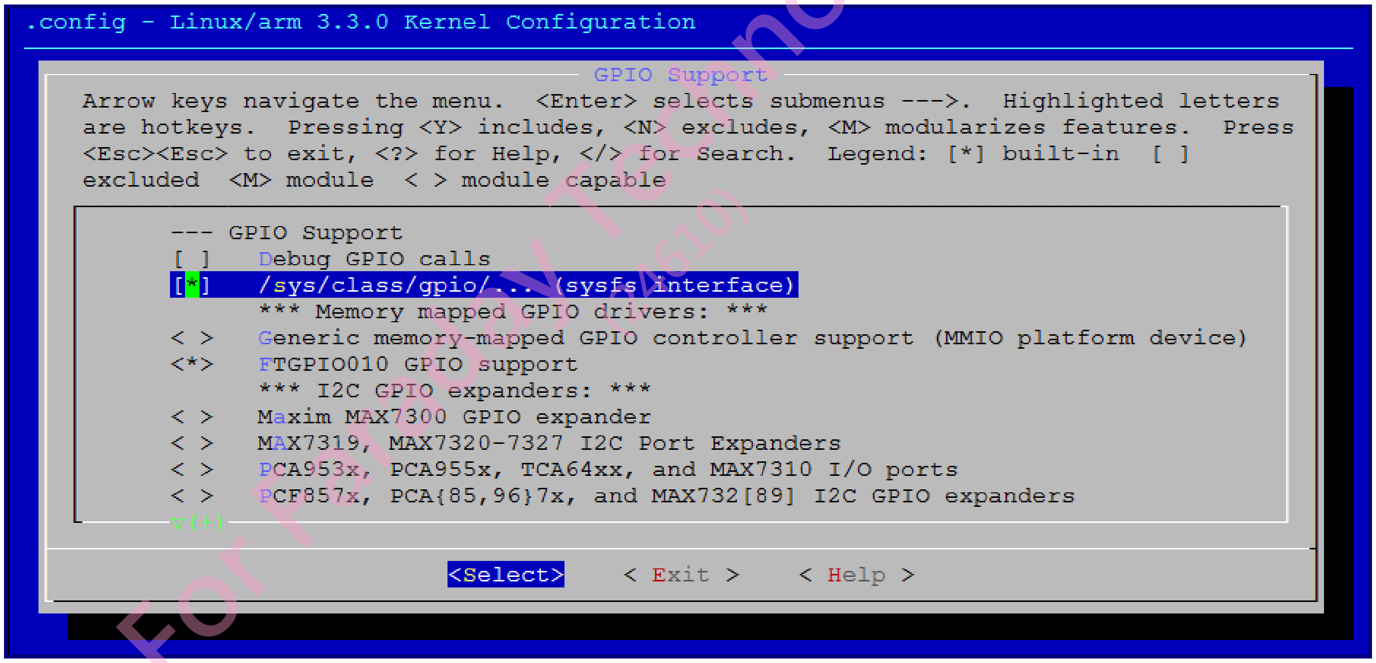
```
static irqreturn_t gm_test_gpio_isr(int irq, void *dev_id)
{
    printk("gpio isr\n");
    gpio_interrupt_clear(0);

    return IRQ_HANDLED;
}

struct fake_dev {
    int foo;
};

static void test_gpio_interrupt(struct fake_dev *dev)
{
    struct gpio_interrupt_mode mode = {
        .trigger_method = GPIO_INT_TRIGGER_EDGE,
        .trigger_edge_nr = GPIO_INT_SINGLE_EDGE,
        .trigger_rise_neg = GPIO_INT_FALLING
    };

    gpio_direction_input(0);
    gpio_interrupt_enable(0);
    gpio_interrupt_setup(0, &mode);

    if (request_irq(gpio_to_irq(0), gm_test_gpio_isr, IRQF_SHARED, "gpio-0", dev) < 0)
    {
        printk("%s fail: request_irq not OK!!\n", __FUNCTION__);
    }
}
```

**Figure 1-4. Example of Using GPIO Input as Interrupt Code**

Please note that users must use gpio_request() to control the pin.
In the test_gpio_interrupt function, Grain Media constructs a mode variable, which specifies that this pin should be triggered from high to low. Then, request_irq() is used to register ISR. Please take the IRQF_SHARED flag into consideration. Please use the GPIO input as the interrupt for all pins of the same GPIO port that have the same IRQ number. Users should use the last parameter in request_irq(), dev, to distinguish the pin from being triggered in ISR.

## 1.4 GPIO in User Space

If users use the Linux GPIO subsystem, GPIO can be set in the user space for kernel sysfs.
First, users should make sure that the following menu configuration is set.
Device Drivers ---> GPIO Support ---> /sys/class/gpio/... (sysfs interface)



```
.config - Linux/arm 3.3.0 Kernel Configuration
                              GPIO Support
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters
  are hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes features.  Press
  <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ]
  excluded  <M> module  < > module capable

        --- GPIO Support
        [ ]   Debug GPIO calls
        [*]   /sys/class/gpio/... (sysfs interface)
              *** Memory mapped GPIO drivers: ***
        < >   Generic memory-mapped GPIO controller support (MMIO platform device)
        <*>   FTGPIO010 GPIO support
              *** I2C GPIO expanders: ***
        < >   Maxim MAX7300 GPIO expander
        < >   MAX7319, MAX7320-7327 I2C Port Expanders
        < >   PCA953x, PCA955x, TCA64xx, and MAX7310 I/O ports
        < >   PCF857x, PCA{85,96}7x, and MAX732[89] I2C GPIO expanders
        v(+)

                  <Select>    < Exit >    < Help >
```

**Figure 1-5. sysfs Setting for GPIO**

After building the kernel image and booting the system, users should follow the steps below for operating GPIO. (The blue words mean results.)

**/ # ls /sys/class/gpio/** <<- To see if GPIO is correct or not

```
export      gpiochip0   gpiochip32  gpiochip64  unexport
/ # echo 0 > /sys/class/gpio/export <<-  Request port0 pin0. (If using port1 pin0, echo 32 >
/sys/class/gpio/export)
/ # ls /sys/class/gpio/ <<- Generate a new folder(gpio0) and its attributes is in the folder
export      gpio0       gpiochip0   gpiochip32  gpiochip64  unexport
/ # cat /sys/class/gpio/gpio0/direction <<- Check input of output
in
/ # cat /sys/class/gpio/gpio0/value <<- Above result is input and check its voltage. 1 means high.
1
/ # echo out > /sys/class/gpio/gpio0/direction <<- Change to output
/ # cat /sys/class/gpio/gpio0/direction <<- Check result
out
/ # cat /sys/class/gpio/gpio0/value  <<- Check voltage of output
0
/ # echo 1 > /sys/class/gpio/gpio0/value <<- Set voltage to high
/ # cat /sys/class/gpio/gpio0/value <<- Check result
1
```

Important: Before using this user space function, users must set **pinmux** for the GPIO pin.

## 1.5 Related File

Documentation/gpio.txt