망고 210 ICS 4.0.4 TVP5150 TV 인코더

Developer Guide

http://www.mangoboard.com/
http://cafe.naver.com/embeddedcrazyboys
Crazy Embedded Laboratory

Document History

Revision	Date	Change note

tvp5150

1. 커널 수정

아래와 같이 변경합니다.

kernel$ vi arch/arm/mach-s5pv210/mach-mango210.c

```
#ifdef CONFIG_VIDEO_TVP5150
#include <media/tvp5150_platform.h>
#endif
……
#ifdef CONFIG_VIDEO_TVP5150
/*
 * Guide for Camera Configuration for mango board
 * ITU CAM CH A: LSI tvp5150
 */
/* External camera module setting */

static DEFINE_MUTEX(tvp5150_lock);

static bool tvp5150_powered_on[2];

#if 1
static inline int tvp5150_get_gpio_stby(int port)
{
	if (port == 0)
		return S5PV210_GPJ4(1);
	else
		return S5PV210_GPJ4(3);
}

static inline int tvp5150_get_gpio_rst(int port)
{
	if (port == 0)
		return S5PV210_GPJ4(2);
	else
		return S5PV210_GPJ4(4);
}
```

```
static inline int tvp5150_get_gpio_mclk(int port)
{
	if (port == 0)
		return S5PV210_GPE1(3);
	else
		return S5PV210_GPJ1(4);
}
static int tvp5150_request_gpio(int port)
{
	int err;
	int gpio_stby, gpio_rst;

	gpio_stby = tvp5150_get_gpio_stby(port);
	gpio_rst = tvp5150_get_gpio_rst(port);

	/* CAM_VGA_nSTBY - GPJ4(1) */
	err = gpio_request(gpio_stby, "TVDEC STBY");
	if (err) {
		pr_err("Failed to request GPIO for camera control₩n");
		return -EINVAL;
	}

	/* CAM_VGA_nRST - GPJ4(2) */
	err = gpio_request(gpio_rst, "TVDEC RESET");
	if (err) {
		pr_err("Failed to request GPIO for camera control₩n");
		return -EINVAL;
	}

	return 0;
}

static int tvp5150_power_on(int port, int on)
{
	int err = 0;
	int gpio_stby, gpio_mclk, gpio_rst;

	gpio_stby = tvp5150_get_gpio_stby(port);
```

```
	gpio_rst = tvp5150_get_gpio_rst(port);
	gpio_mclk = tvp5150_get_gpio_mclk(port);

	if (on) {
		/* CAM_VGA_nSTBY HIGH */
		gpio_direction_output(gpio_stby, 0);
		gpio_set_value(gpio_stby, 1);
		udelay(10);

		/* Mclk enable */
		s3c_gpio_cfgpin(gpio_mclk, S3C_GPIO_SFN(0x02));
		udelay(430);

		/* CAM_VGA_nRST HIGH */
		gpio_direction_output(gpio_rst, 0);
		gpio_set_value(gpio_rst, 1);
		mdelay(5);
	} else {
			gpio_direction_output(gpio_rst, 1);
		gpio_set_value(gpio_rst, 0);
		udelay(430);

		/* Mclk disable */
		s3c_gpio_cfgpin(gpio_mclk, 0);
		udelay(1);

		/* CAM_VGA_nSTBY LOW */
		gpio_direction_output(gpio_stby, 1);
		gpio_set_value(gpio_stby, 0);
	}

	return 0;
}

static int tvp5150_port_a_power_en(int onoff)
{
	int err = 0;
	mutex_lock(&tvp5150_lock);
```

```
	/* we can be asked to turn off even if we never were turned
	 * on if something odd happens and we are closed
	 * by camera framework before we even completely opened.
	 */
	if (onoff != tvp5150_powered_on[0]) {
		err = tvp5150_power_on(0, 0);
		err = tvp5150_power_on(0, 1);
		if (!err)
			tvp5150_powered_on[0] = onoff;
	}
	mutex_unlock(&tvp5150_lock);

	return err;
}

static int tvp5150_port_b_power_en(int onoff)
{
	int err = 0;

	mutex_lock(&tvp5150_lock);
	/* we can be asked to turn off even if we never were turned
	 * on if something odd happens and we are closed
	 * by camera framework before we even completely opened.
	 */
	if (onoff != tvp5150_powered_on[1]) {
		err = tvp5150_power_on(1, 0);//CRZ bug
		err = tvp5150_power_on(1, 1);// CRZ
		if (!err)
			tvp5150_powered_on[1] = onoff;
	}
	mutex_unlock(&tvp5150_lock);

	return err;
}

#else

#endif
```

```
static struct tvp5150_platform_data tvp5150_a_plat = {
        .default_width = 720,
        .default_height = 525,
        .pixelformat = V4L2_PIX_FMT_YUYV,
        .freq = 27000000,
        .is_mipi = 0,

        .cam_power = tvp5150_port_a_power_en,
};

static struct tvp5150_platform_data tvp5150_b_plat = {
        .default_width = 720,
        .default_height = 525,
        .pixelformat = V4L2_PIX_FMT_YUYV,
        .freq = 27000000,
        .is_mipi = 0,

        .cam_power = tvp5150_port_b_power_en,
};


static struct i2c_board_info tvp5150_a_i2c_info = {
        I2C_BOARD_INFO("tvp5150", 0xb8>>1),
        .platform_data = &tvp5150_a_plat,
};

static struct i2c_board_info tvp5150_b_i2c_info = {
        I2C_BOARD_INFO("tvp5150", 0xb8>>1),
        .platform_data = &tvp5150_b_plat,
};

static struct s3c_platform_camera tvp5150_a = {
        .id             = CAMERA_PAR_A,
        .type           = CAM_TYPE_ITU,
        .fmt            = ITU_656_YCBCR422_8BIT,
        .order422       = CAM_ORDER422_8BIT_CBYCRY,
        .i2c_busnum     = 0,
```

```
		.info 		= &tvp5150_a_i2c_info,
		.pixelformat	= V4L2_PIX_FMT_UYVY,
		.srclk_name	= "xusbxti",
		.clk_name	= "sclk_cam0",
		.clk_rate	= 27000000,
		.line_length	= 525,
		.width		= 720,
		.height		= 525,
		.window		= {
			.left	= 0,
			.top	= 0,
			.width	= 720,
			.height = 525,
		},

		/* Polarity */
		.inv_pclk	= 0,
		.inv_vsync	= 0,
		.inv_href	= 0,
		.inv_hsync	= 0,

		.initialized	= 0,
		.cam_power	= tvp5150_port_a_power_en,
};

static struct s3c_platform_camera tvp5150_b = {
		.id		= CAMERA_PAR_B,
		.type		= CAM_TYPE_ITU,
		.fmt		= ITU_656_YCBCR422_8BIT,
		.order422	= CAM_ORDER422_8BIT_CBYCRY,
		.i2c_busnum	= 1,
		.info		= &tvp5150_b_i2c_info,
		.pixelformat	= V4L2_PIX_FMT_UYVY,
		.srclk_name	= "xusbxti",
		.clk_name	= "sclk_cam1",
		.clk_rate	= 27000000,
		.line_length	= 525,
		.width		= 720,
```

```
		.height		= 525,
		.window		= {
			.left	= 0,
			.top	= 0,
			.width	= 720,
static struct s3c_platform_camera tvp5150_b = {
		.id		= CAMERA_PAR_B,
		.type		= CAM_TYPE_ITU,
		.fmt		= ITU_656_YCBCR422_8BIT,
		.order422	= CAM_ORDER422_8BIT_CBYCRY,
		.i2c_busnum	= 1,
		.info		= &tvp5150_b_i2c_info,
		.pixelformat	= V4L2_PIX_FMT_UYVY,
		.srclk_name	= "xusbxti",
		.clk_name	= "sclk_cam1",
		.clk_rate	= 27000000,
		.line_length	= 525,
		.width		= 720,
		.height		= 525,
		.window		= {
			.left	= 0,
			.top	= 0,
			.width	= 720,
			.height = 525,
		},

		/* Polarity */
		.inv_pclk	= 0,
		.inv_vsync	= 0,
		.inv_href	= 0,
		.inv_hsync	= 0,

		.initialized	= 0,
		.cam_power	= tvp5150_port_b_power_en,
};
```

```
#endif



#ifdef CONFIG_VIDEO_TVP5150
        tvp5150_request_gpio(0);
        tvp5150_request_gpio(1);
        tvp5150_port_a_power_en(1); //CRZ sda pull down bug
        tvp5150_port_b_power_en(1); //CRZ sda pull down bug
#endif
……..
#ifdef CONFIG_VIDEO_TVP5150
                &tvp5150_a,
                &tvp5150_b,
#endif
```

kernel$ vi drivers/media/video/Kconfig

```
config VIDEO_TVP5150
        tristate "Texas Instruments TVP5150 video decoder"
        depends on VIDEO_V4L2 && I2C
        ---help---
          Support for the Texas Instruments TVP5150 video decoder.

          To compile this driver as a module, choose M here: the
          module will be called tvp5150.
```

include/media/tvp5150_platform.h

drivers/media/video/tvp5150.c

위의 경로의 소스를 사용합니다.

make파일을 확인합니다.

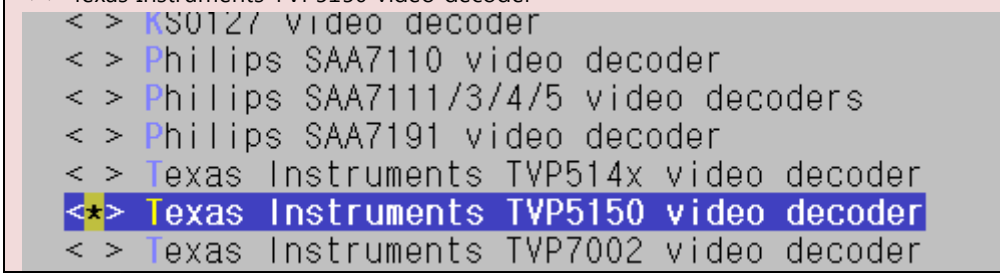
/kernel/drivers$ vi media/video/Makefile

```
obj-$(CONFIG_VIDEO_VINO) += indycam.o
obj-$(CONFIG_VIDEO_TVP5150) += tvp5150.o
obj-$(CONFIG_VIDEO_TVP514X) += tvp514x.o
```

2. 커널 빌드

./build_kernel config

```
Device Drivers  --->
<*> Multimedia support  --->
[*]   Video capture adapters  --->
< >   SR130PC10 Camera Sensor
Encoders, decoders, sensors and other helper chips  --->
<*> Texas Instruments TVP5150 video decoder
```




```
cp .config mango210_10.1inch_tvp5150_defconfig
cp mango210_10.1inch_ tvp5150_defconfig  arch/arm/configs/
./build_kernel defconfig mango210_10.1inch_ tvp5150_defconfig
./build_kernel
```

zImage가 만들어집니다.

만들어진 zImage는 /kernel/arch/arm/boot에 있고 image에도 생성됩니다.