How To Debug SPI

Revision	Date	Description
V1.0	2016/06/30	Initial Draft

Archermind

2016/7/19

Contents

1. Introduction

In android M version, the SPI usage are the same as L, the SPI parameters are set from SPI slave device driver, only padmacro customization in DTS.

2. SPI Parameters Patch and Set

Patch:kernel-3.18/arch/arm64/boot/dts/mt6797.dtsi

```
spi0:spi@1100a000 {
                compatible = "mediatek,mt6797-spi";
                cell-index = <0>;
                spi-padmacro = <0>;
                reg = <0x1100a000 0x1000>;
                interrupts = <GIC_SPI 122 IRQ_TYPE_LEVEL_LOW>;
                clocks = <&infrasys INFRA_SPI>;
                clock-names = "spi-main";
                clock-frequency = <109000000>;
                clock-div = <1>;
        };

spi1:spi@11012000 {
        compatible = "mediatek,mt6797-spi";
        cell-index = <1>;
        spi-padmacro = <1>;
        reg = <0x11012000 0x1000>;
        interrupts = <GIC_SPI 131 IRQ_TYPE_LEVEL_LOW>;
        clocks = <&infrasys INFRA_SPI1>;
        clock-names = "spi-main";
        clock-frequency = <109000000>;
        clock-div = <1>;
        };

spi2:spi@11018000 {
        compatible = "mediatek,mt6797-spi";
        cell-index = <2>;
        spi-padmacro = <1>;
        reg = <0x11018000 0x1000>;
        interrupts = <GIC_SPI 132 IRQ_TYPE_LEVEL_LOW>;
        clocks = <&infrasys INFRA_SPI2>;
        clock-names = "spi-main";
        clock-frequency = <109000000>;
        clock-div = <1>;
        };

spi3:spi@11019000 {
        compatible = "mediatek,mt6797-spi";
```

```
        cell-index = <3>;
        spi-padmacro = <0>;
        reg = <0x11019000 0x1000>;
        interrupts = <GIC_SPI 133 IRQ_TYPE_LEVEL_LOW>;
        clocks = <&infrasys INFRA_SPI3>;
        clock-names = "spi-main";
        clock-frequency = <109000000>;
        clock-div = <1>;
        };

spi4:spi@1101a000 {
        compatible = "mediatek,mt6797-spi";
        cell-index = <4>;
        spi-padmacro = <0>;
        reg = <0x1101a000 0x1000>;
        interrupts = <GIC_SPI 134 IRQ_TYPE_LEVEL_LOW>;
        clocks = <&infrasys INFRA_SPI4>;
        clock-names = "spi-main";
        clock-frequency = <109000000>;
        clock-div = <1>;
        };

spi5:spi@1101b000 {
        compatible = "mediatek,mt6797-spi";
        cell-index = <5>;
        spi-padmacro = <0>;
        reg = <0x1101b000 0x1000>;
        interrupts = <GIC_SPI 135 IRQ_TYPE_LEVEL_LOW>;
        clocks = <&infrasys INFRA_SPI5>;
        clock-names = "spi-main";
        clock-frequency = <109000000>;
        clock-div = <1>;
        };
```

3. Configuration

3.1. Project configuration

Path:alps/kernel-3.18/arch/arm64/configs/(&project)_defconfig

CONFIG_MTK_SPI=y

3.2. Use dct_toot Config Pin

242	GPIO242		1:SPI2_CLK_B	☑	☑	☑	☑	☑	☑	☑	☑	☑	☐	IN	☑	☑	☐	GPIO_SPI2_SCK_PIN
243	GPIO243		1:SPI2_MI_B	☑	☑	☑	☑	☑	☑	☑	☑	☑	☐	IN	☑	☑	☐	GPIO_SPI2_MISO_PIN
244	GPIO244		1:SPI2_MO_B	☑	☑	☑	☑	☑	☑	☑	☑	☑	☐	IN	☑	☑	☐	GPIO_SPI2_MOSI_PIN
245	GPIO245		1:SPI2_CS_B	☑	☑	☑	☑	☑	☑	☑	☑	☑	☐	IN	☑	☑	☐	GPIO_SPI2_CS_PIN

eint: #155

3.3. Driver Code

Please refer to: alps/kernel-3.18/drivers/spi/mediatek/mt6797/spidev.c

3.4. Register Dev

Please refer to mt6797.dtsi

3.5. Register Drv

```
static const struct of_device_id mt_spi_of_match[] = {
        {.compatible = "mediatek,mt6735-spi",},
        {.compatible = "mediatek,mt6735m-spi",},
        {.compatible = "mediatek,mt6753-spi",},
        {.compatible = "mediatek,mt6797-spi",},
        {},
};

MODULE_DEVICE_TABLE(of, mt_spi_of_match);

struct platform_driver mt_spi_driver = {
        .driver = {
```

```
                .name = "mt-spi",
                .owner = THIS_MODULE,
                .of_match_table = mt_spi_of_match,
                },
        .probe = mt_spi_probe,
        .suspend = mt_spi_suspend,
        .resume = mt_spi_resume,
        .remove = __exit_p(mt_spi_remove),
};

static int __init mt_spi_init(void)
{
        int ret;

        pr_debug("SPI init!\n");
        ret = platform_driver_register(&mt_spi_driver);
        return ret;
}
```

4. Transfer Data API

Please refer to API:

```
spi_setup_xfer(&spi->dev, &test_xfer, len, 1);
spi_message_add_tail(&test_xfer, &msg);
ret = spi_sync(spi, &msg);
```

5. Parameter Configuration

we use spi_device->controller_data pass parameter structure to controller.
– others configuration is coherent with your device such as CPOL, Setup time, please refer to your datasheet.

```
struct mt_chip_conf {
        u32 setuptime;
        u32 holdtime;
        u32 high_time;
        u32 low_time;
        u32 cs_idletime;
        u32 ulthgh_thrsh;
        enum spi_sample_sel sample_sel;
        enum spi_cs_pol cs_pol;
        enum spi_cpol cpol;
        enum spi_cpha cpha;
        enum spi_mlsb tx_mlsb;
        enum spi_mlsb rx_mlsb;
        enum spi_endian tx_endian;
        enum spi_endian rx_endian;
        enum spi_transfer_mode com_mod;
        enum spi_pause_mode pause;
        enum spi_finish_intr finish_intr;
        enum spi_deassert_mode deassert;
        enum spi_ulthigh ulthigh;
        enum spi_tckdly tckdly;
};

  struct mt_chip_conf *spi_par;

        pr_debug("%s\n", __func__);

        fpc1020->spi->controller_data = (void *)&spi_conf;
        spi_par = &spi_conf;

        spi_par->setuptime = 20;
        spi_par->holdtime = 20;
        spi_par->high_time = 50;
        spi_par->low_time = 50;
        spi_par->cs_idletime = 5;
        spi_par->rx_mlsb = 1;
        spi_par->tx_mlsb = 1;
```

```
spi_par->tx_endian = 0;
spi_par->rx_endian = 0;
spi_par->cpol = 0;
spi_par->cpha = 0;
spi_par->com_mod = FIFO_TRANSFER;
spi_par->pause = 1;
spi_par->finish_intr = 1;
spi_par->deassert = 0;

fpc1020->spi->mode = SPI_MODE_0;
fpc1020->spi->bits_per_word = 8;
fpc1020->spi->chip_select = 0;

error = spi_setup(fpc1020->spi);
```