时钟流程:

ddr\_pwr\_pll\_mclk parent\_rate:26000000   --->晶振26M  ddr\_pwr\_pll\_mclk的父时钟osc\_clk=26M

ddr\_pwr\_pll\_mclk rate:1066000000   ---> ddr\_pwr\_pll\_mclk 通过osc\_clk分频后的时钟

    --->clk->rate = parent\_rate / refdiv / postdiv1 / postdiv2 \* fbdiv; = 1066000000

sdiv\_init com\_pre\_clk rate 213200000     --->   com\_pre\_clk通过父时钟   ddr\_pwr\_pll\_mclk(1066000000)分频得到的时钟

        --->clk->rate = clk->parent->rate / (div + 1) =  1066000000/(4 + 1)  = 1066000000/5 = 213200000

fdiv\_init\_by\_table pcm\_clk rate 51448 ---> pcm\_clk的默认时钟51448

pcm\_clk parent\_rate:213200000

---->

clk\_enable\_generic pcm\_clk rate:25600000

pcm\_clk parent\_rate:213200000

static struct clk com\_pcm\_clk = {

    .name = "pcm\_clk",

    .parent = &com\_pre\_clk,

    .cust = &clk\_com\_pcm\_table,

    .mclk\_reg = (void \_\_iomem \*)io\_p2v(DDR\_PWR\_CLKEN0),//0xE100A03c

            --->val = (1 << clk->mclk\_bit) | (1 << clk->mclk\_we\_bit) | clk->divclk\_val;

    .mclk\_bit = 4,//

    .mclk\_we\_bit = 20,

    .ifclk\_reg = (void \_\_iomem \*)io\_p2v(DDR\_PWR\_PCLKEN),

                --->val = (1 << clk->ifclk\_bit) | (1 << clk->ifclk\_we\_bit);

                --->writel(val, clk->ifclk\_reg);

    .ifclk\_bit = 4,

    .ifclk\_we\_bit = 20,

    .divclk\_reg = (void \_\_iomem \*)io\_p2v(DDR\_PWR\_PCMCLKDIV),

        --->fdiv\_set\_rate\_by\_table

            --->writel(((tbl->mul << 16) | tbl->div), clk->divclk\_reg);

    .init = &fdiv\_init\_by\_table,

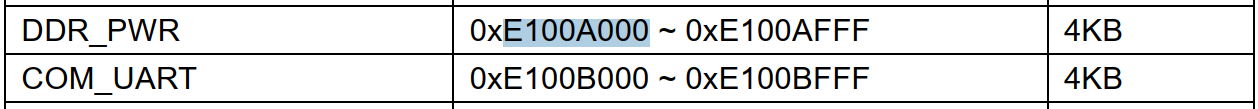
    .enable = &clk\_enable\_generic,

    .disable = &clk\_disable\_generic,

    .set\_rate = &fdiv\_set\_rate\_by\_table,

};

时钟：



lc1860 设备树 pcm clk

#define PWR\_MASK(len) ((1 << len) - 1)

#define PWR\_BIT\_SFT(x)  (x)

#define PWR\_WEBIT\_SFT(x,offset) (PWR\_BIT\_SFT(x) + offset)

#define PWR\_WEBIT\_DEF(x) PWR\_WEBIT\_SFT(x,16)

#define PWR\_WEBIT\_NONE 0

#define DDR\_PWR\_BASE        (0xE100A000)

#define DDR\_PWR\_PCLKEN                    (DDR\_PWR\_BASE + 0x3c)

#define DDR\_PWR\_CLKEN0                    (DDR\_PWR\_BASE + 0x40)

#define DDR\_PWR\_PCMCLKDIV                (0x20)

com\_pcm\_clkgt: com\_pcm\_clkgt {

                                compatible = "comip,clk-gate";

                                                        --->CLK\_OF\_DECLARE(comip\_gate, "comip,clk-gate", comip\_clkgate\_setup)

                                #clock-cells = <0>;

                                clocks = <&com\_pre\_clk>;

                                comip,clkgt = <DDR\_PWR\_CLKEN0 PWR\_BIT\_SFT(4) PWR\_WEBIT\_DEF(4)>;---><0xE100A040  4    20 > ---> <gclk->reg     gclk->shift    gclk->weshift>

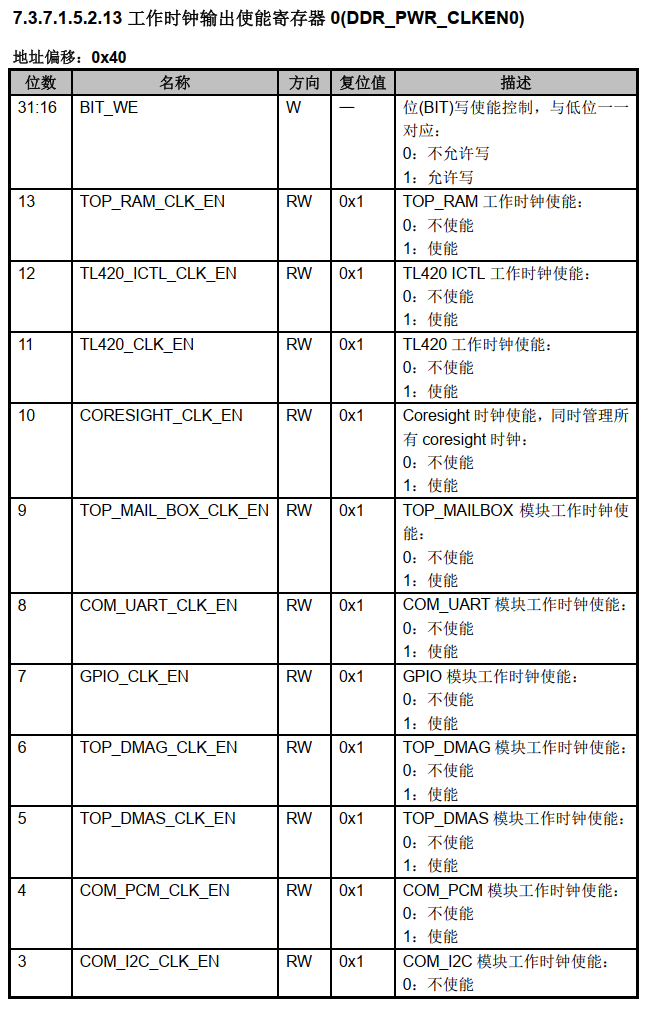
                                <时钟寄存器    时钟寄存器使能位   时钟寄存器使能位的位使能控制 >

                                --->val = readl(gclk->reg);

                                --->val |= (1 << gclk->shift);

                                --->val |= (1 << gclk->weshift);

                                --->writel(val, gclk->reg);



                                clock-output-names = "pcm\_clkgt";

                        };

                        com\_pcm\_clk: com\_pcm\_clk {

                                compatible = "comip,clk-fdiv";

                                                        --->CLK\_OF\_DECLARE(comip\_fdiv, "comip,clk-fdiv", comip\_clkfdiv\_setup)

                                                                --->fdclk->table = clk\_com\_pcm\_table;

                                                                                               ---->struct comip\_fdiv\_table clk\_com\_pcm\_table[] = {

    {64000,    265200000, 8,    16575},

    {64000,    213200000, 5,    8328},

    {64000,    208000000, 1,    1625},

    {25600000, 213200000, 3980, 16575},

    {51448,    213200000, 8,    16575},//boot

    {0, 0, 0, 0}

};                                                            ---->.set\_rate = comip\_clkfdiv\_set\_rate,

                                                                                        --->

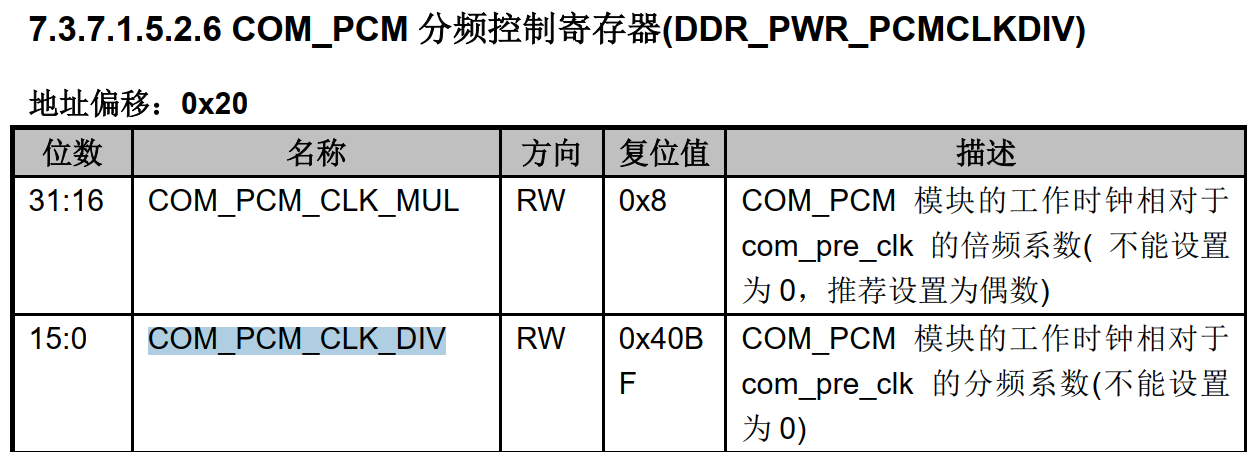
                                #clock-cells = <0>;

                                clocks = <&com\_pcm\_clkgt>;

                                comip,clkfdiv = <DDR\_PWR\_PCMCLKDIV PWR\_MASK(16) PWR\_BIT\_SFT(0) PWR\_MASK(16) PWR\_BIT\_SFT(16)>;

                                val = ((clkft->div & fdclk->divmask) << fdclk->divshift) |

                                        ((clkft->mul & fdclk->mulmask) << fdclk->mulshift);



                                clk\_com\_pcm\_table;

                                clock-output-names = "com\_pcm\_clk";

                        };

                        com\_pcm\_pclkgt: com\_pcm\_pclkgt {

                                compatible = "comip,clk-gate";

                                #clock-cells = <0>;

                                clocks = <&com\_apb\_clk>;

                                comip,clkgt = <DDR\_PWR\_PCLKEN PWR\_BIT\_SFT(4) PWR\_WEBIT\_DEF(4)>;---><0xE100A03c  4    20 > ---> <gclk->reg     gclk->shift    gclk->weshift>



                                clock-output-names = "com\_pcm\_pclkgt";

                        };

----->

  com\_pcm: pcm@E1007000 {

                        compatible = "leadcore,comip-pcm";

                        reg = <0xE1007000 0x400>;

                        rx\_dma\_channel = <&top\_dmas 10 0>;

                        tx\_dma\_channel = <&top\_dmas 2 0>;

                        clocks = <&com\_pcm\_clk>, <&com\_pcm\_pclkgt>;

                        clock-names = "pcm\_clk", "pcm\_pclk";

                };

    ---->comip\_pcm\_startup

        --->pcm->clk = clk\_get(dai->dev, "pcm\_clk");

        --->pcm->pclk = clk\_get(dai->dev, "pcm\_pclk");

        --->clk\_set\_rate(pcm->clk, COMIP\_PCM\_CLK\_RATE);

        --->clk\_prepare\_enable(pcm->clk);

        --->clk\_prepare\_enable(pcm->pclk);

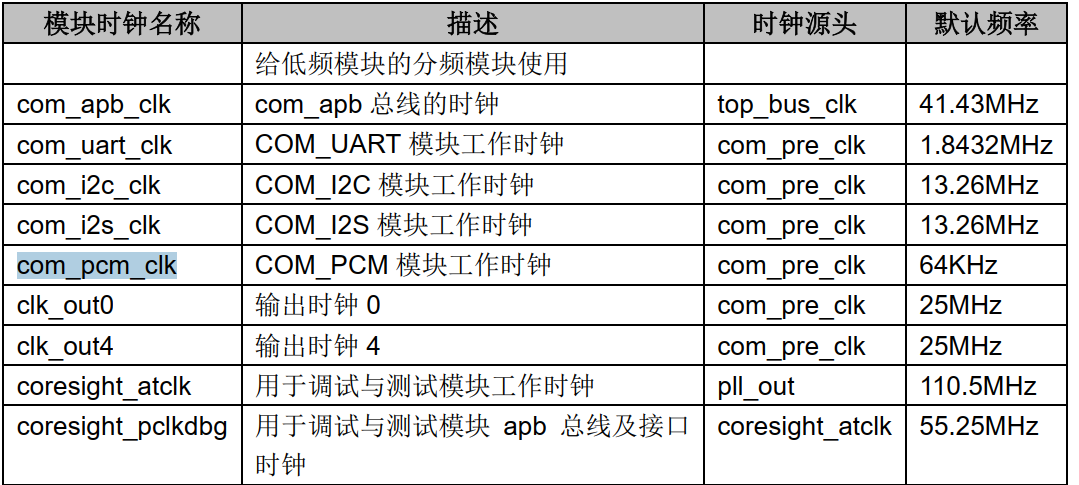
1:获取pcm的mclk 寄存器 DDR\_PWR\_OCMCLKDIV

comip\_pcm\_startup

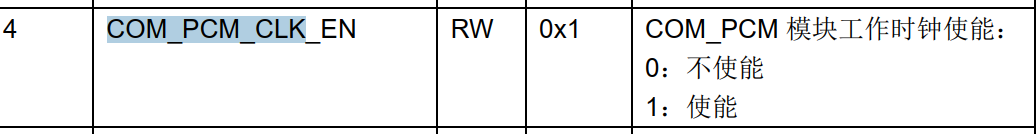
    --->pcm->clk = clk\_get(dai->dev, "pcm\_clk");

    --->clk\_set\_rate(pcm->clk, COMIP\_PCM\_CLK\_RATE);

    --->clk\_enable(pcm->clk);



使能时钟:



2: pcm sclk 时钟

