

# Rockchip Gpio Output Clocks

---

文件标识：RK-KF-YF-486

发布版本：V1.2.0

日期：2023-02-27

文件密级：☐绝密 ☐秘密 ☐内部资料 ☒公开

## 免责声明

本文档按“现状”提供，瑞芯微电子股份有限公司（“本公司”，下同）不对本文档的任何陈述、信息和内容的准确性、可靠性、完整性、适销性、特定目的性和非侵权性提供任何明示或暗示的声明或保证。本文档仅作为使用指导的参考。

由于产品版本升级或其他原因，本文档将可能在未经任何通知的情况下，不定期进行更新或修改。

## 商标声明

“Rockchip”、“瑞芯微”、“瑞芯”均为本公司的注册商标，归本公司所有。

本文档可能提及的其他所有注册商标或商标，由其各自拥有者所有。

## 版权所有© 2023 瑞芯微电子股份有限公司

超越合理使用范畴，非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

瑞芯微电子股份有限公司

Rockchip Electronics Co., Ltd.

地址：福建省福州市铜盘路软件园A区18号

网址：[www.rock-chips.com](http://www.rock-chips.com)

客户服务电话：+86-4007-700-590

客户服务传真：+86-591-83951833

客户服务邮箱：[fae@rock-chips.com](mailto:fae@rock-chips.com)

前言

概述

产品版本

| 芯片名称     | 内核版本         |
|----------|--------------|
| RK3399   | linux4.4及以上  |
| RK1808   | linux4.4及以上  |
| RK3328   | linux4.4及以上  |
| RK3308   | linux4.4及以上  |
| RV1126   | linux4.19及以上 |
| PX30     | linux4.4及以上  |
| RK3566/8 | linux4.19及以上 |
| RK3588   | linux5.10    |
| RK3528   | linux4.19    |

读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师

软件开发工程师

修订记录

| 版本号    | 作者 | 修改日期       | 修改说明       |
|--------|----|------------|------------|
| V1.0.0 | 张晴 | 2021-09-06 | 第一次版本发布    |
| V1.1.0 | 张晴 | 2021-12-29 | 增加RK3588支持 |
| V1.2.0 | 张晴 | 2023-02-27 | 增加RK3528支持 |

# 目录

## Rockchip Gpio Output Clocks

1. 概念介绍
  - 1.1 概念解释
  - 1.2 主要配置
  - 1.3 GPIO和CLK对应关系
  - 1.4 IO命令示例
  - 1.5 GPIO输出CLK软件修改

# 1. 概念介绍

---

## 1.1 概念解释

随着外围设备增加以及成本约束，很多外设会希望使用SOC的IO输出某些特定时钟频率来替换外部晶振，从而减少外围电路的成本。

## 1.2 主要配置

SOC上IO输出CLK，前提是这个IO的IOMUX有CLK输出功能，

### 1. IOMUX

这个IO的IOMUX必须配置成CLK输出功能。

### 2. CLK频率

需要配置CLK频率。因为是替代外部晶振的，所有这个CLK如果是输出24M并且是从SOC的晶振bypass的，那么信号质量最好。如果是从PLL分频下来的信号质量稍微差一些，是否可以满足外设需求要看外设的参考文档。

### 2. IO驱动能力

有的IO上测试到的波形并不好看，可能还需要调整IO的驱动能力。

## 1.3 GPIO和CLK对应关系

| 芯片名称     | clk name              | gpio name         | 支持频率             |
|----------|-----------------------|-------------------|------------------|
| RK3288   | SCLK_TESTOUT          | GPIO0_C1          | 24M、32K          |
| RK3328   | SCLK_TESTOUT          | GPIO1_A6          | 24M、32K          |
| PX30     | SCLK_TEST_OUT         | GPIO1_D6\GPIO0_B5 | 24M、32K          |
| RK3308   | SCLK_TESTOUT          | GPIO0_A4          | 24M、32K          |
| RV1108   | SCLK_TESTOUT          | GPIO0_A6          | 24M、32K          |
| RV1126   | SCLK_TESTOUT          | GPIO1_A4          | 24M、32K          |
| RK3368   | SCLK_TESTOUT          | GPIO0_B0          | 24M、32K          |
| RK3399   | SCLK_TESTOUT          | GPIO2_D1\GPIO0_B0 | 24M、32K          |
| RK3566/8 | CLK_WIFI              | GPIO0_A0          | 24M              |
| RK3566/8 | CLK_MAC0_OUT          | GPIO2_C1          | 24M、25M、50M、125M |
| RK3566/8 | CLK_MAC1_OUT          | GPIO3_B0          | 24M、25M、50M、125M |
| RK3566/8 | CLK_MAC1_OUT          | GPIO4_B3          | 24M、25M、50M、125M |
| RK3566/8 | CLK_CIF_OUT           | GPIO4_C0          | 24M、27M、37.125M  |
| RK3566/8 | CLK_CAM0_OUT          | GPIO4_A7          | 24M、27M、37.125M  |
| RK3588   | REFCLKOUT             | GPIO0_A0          | 24M              |
| RK3588   | CLK_GMAC_50M          | GPIO4_C3          | 50M、125M         |
| RK3588   | REFCLK25M_ETH0        | GPIO4_B0          | 25M、50M          |
| RK3588   | CLK_DEEPSLOW          | GPIO2_C5          | 32K              |
| RK3588   | CLK_MIPI_CAMARAOUT_M0 | GPIO4_B1          | 24M、27M、37.125M  |
| RK3588   | CLK_MIPI_CAMARAOUT_M1 | GPIO1_B6          | 24M、27M、37.125M  |
| RK3588   | CLK_MIPI_CAMARAOUT_M2 | GPIO1_B7          | 24M、27M、37.125M  |
| RK3588   | CLK_MIPI_CAMARAOUT_M3 | GPIO1_D6          | 24M、27M、37.125M  |
| RK3588   | CLK_MIPI_CAMARAOUT_M4 | GPIO1_D7          | 24M、27M、37.125M  |
| RK3588   | CLK_CIF_OUT           | GPIO4_B4          | 24M、27M、37.125M  |
| RK3588   | REFCLK25M_ETH1        | GPIO3_A6          | 25M、50M          |
| RK3528   | CLK_REFOUT            | GPIO0_A1          | 24M              |
| RK3528   | CLK_REFOUT            | GPIO3_C3          | 24M              |
| RK3528   | CLK_DEEPSLOW          | GPIO3_C3          | 32K              |

| 芯片名称   | clk name     | gpio name | 支持频率   |
|--------|--------------|-----------|--------|
| RK3528 | CLK_DEEPSLOW | GPIO1_C3  | 32K    |
| RK3528 | CLK_TESTOUT  | GPIO2_A5  | 要看实际配置 |

## 1.4 IO命令示例

1. RK3288, Test\_clk 输出24M, 测试点GPIO0\_C1

```
io -4 0xff760068 0x0f000000 # test clk div set 1
io -4 0xff760170 0x80000000 # enable test clk
io -4 0xff7601e8 0x0f000800 # test clk select 24m
io -4 0xff73008c 0x000c0005 # gpio0_c1 iomux select testclk
```

2. RK3328, Test\_clk 输出24M, 测试点GPIO1\_A6

```
io -4 0xff440108 0x000f0000 # test clk div set 1
io -4 0xff440200 0x02000000 # enable test clk
io -4 0xff100010 0x30002000 # gpio1_a6 iomux select testclk
io -4 0xff440084 0x1f001700 # test clk select 24m
```

3. PX30, Test\_clk 输出24M, 测试点GPIO1\_D6或者GPIO0\_B5

```
io -4 0xff2b0240 0x80000000 # enable test clk
io -4 0xff2b01e4 0x1f1f0007 # test clk select 24m and div set 1
io -4 0xff14001c 0x0f000300 # gpio1_d6 iomux select testclk
io -4 0xff010004 0x0c000800 # gpio0_b5 iomux select testclk
```

4. RK3308, Test\_clk 输出24M, 测试点GPIO0\_A4

```
io -4 0xff500224 0x1fff1700 # test clk select 24m and div set 1
io -4 0xff500310 0x02000000 # enable test clk
io -4 0xff000000 0x03000100 # gpio0_a4 iomux select testclk
```

5. RV110X, Test\_clk 输出24M, 测试点GPIO0\_A6

```
io -4 0x202000fc 0x1f000000 # test clk div set 1
io -4 0x20200144 0x02000000 # enable test clk
io -4 0x202001cc 0x0f000800 # test clk select 24m
io -4 0x20060000 0x30002000 # gpio0_a6 iomux select testclk
```

6. RV1126, Test\_clk 输出24M, 测试点GPIO1\_A4

```
io -4 0xff49022c 0xffff0000 # test clk select 24m and div set 1
io -4 0xff4902dc 0x20000000 # enable test clk
io -4 0xfe010014 0x00070002 # gpio1_a4 iomux select testclk
```

7. RK3368, Test\_clk 输出24M, 测试点GPIO0\_B0

```
io -4 0xff7601bc 0x1f000000 # test clk div set 1
io -4 0xff76021c 0x00020000 # enable test clk
io -4 0xff760380 0x000f0008 # test clk select 24m
io -4 0xff738004 0x00030001 # gpio0_b0 iomux select testclk
```

8. RK3399, Test\_clk 输出24M, 测试点GPIO2\_D1或者GPIO0\_B0

```
echo 24000000 > d/clk/clk_testout1_pll_src/clk_rate
echo 24000000 > d/clk/clk_testout1/clk_rate # test clk set 24m
echo 1 > d/clk/clk_testout1/clk_enable_count # enable test clk
io -4 0xff77e004 0x000c0008 # gpio2_d1 iomux select testclk

echo 24000000 > d/clk/clk_testout2_pll_src/clk_rate
echo 24000000 > d/clk/clk_testout2/clk_rate # test clk set 24m
echo 1 > d/clk/clk_testout2/clk_enable_count # enable test clk
io -4 0xff320004 0x00030003 # gpio0_b0 iomux select testclk
```

9. RK3566/8, IO输出CLK功能更加强大

(1) CLK32K\_OUT0 测试点gpio0\_b0

```
io -4 0xfdd00100 0x00c00080
io -4 0xfdc20100 0x00010000
io -4 0xfdc20008 0x00030002
```

(2) CLK32K\_OUT1 测试点gpio2\_c6

```
io -4 0xfdd00100 0x00c00080
io -4 0xfdc20100 0x00010000
io -4 0xfdc20008 0x00030002
io -4 0xfdc60034 0x07000100
```

(3) REF\_CLKOUT 测试点gpio0\_a0

```
io -4 0xfdc20000 0x00070001
echo 24000000 >/sys/kernel/debug/clk/clk_wifi/clk_rate
echo 1 > /sys/kernel/debug/clk/clk_wifi/clk_enable_count
```

(4) ETH\_REFCLK\_25M 测试点gpio2\_c1

```
io -4 0xfdc60030 0x00700020
echo 25000000 >/sys/kernel/debug/clk/clk_mac0_out/clk_rate
echo 1 > /sys/kernel/debug/clk/clk_mac0_out/clk_enable_count
```

(5) ETH\_REFCLK\_25M\_M0 测试点gpio3\_b0

```
io -4 0xfdc60048 0x00070003
echo 25000000 >/sys/kernel/debug/clk/clk_mac1_out/clk_rate
echo 1 > /sys/kernel/debug/clk/clk_mac1_out/clk_enable_count
```

(6) ETH\_REFCLK\_25\_M1 测试点gpio4\_b3

```
io -4 0xfdc60068 0x70003000
echo 25000000 >/sys/kernel/debug/clk/clk_mac1_out/clk_rate
echo 1 > /sys/kernel/debug/clk/clk_mac1_out/clk_enable_count
```

(7) CIF\_CLKOUT 测试点gpio4\_c0

```
io -4 0xfdc60070 0x00070001
echo 27000000 >/sys/kernel/debug/clk/clk_cif_out/clk_rate
echo 1 > /sys/kernel/debug/clk/clk_cif_out/clk_enable_count
```

(8) CAM\_CLKOUT0 测试点gpio4\_a7

```
io -4 0xfdc60064 0x70001000
echo 27000000 >/sys/kernel/debug/clk/clk_cam0_out/clk_rate
echo 1 > /sys/kernel/debug/clk/clk_cam0_out/clk_enable_count
```

(9) CAM\_CLKOUT1 测试点gpio4\_b0

```
io -4 0xfdc60068 0x00070001
echo 24000000 >/sys/kernel/debug/clk/clk_cam1_out/clk_rate
echo 1 > /sys/kernel/debug/clk/clk_cam1_out/clk_enable_count
```

(10) TESTCLK 测试点gpio2\_a2

```
io -4 0xfdc60020 0x07000200
io -4 0xfdd20228 0x1fff0000
io -4 0xfdd20388 0x80000000
```

备注：如果想测试其他频率或者频率点，按照上面寄存器查找TRM，有详细说明。

10. RK3588, IO输出CLK功能更加强大

1)GPIO0\_A0 -> refclkout 24M xin\_osc0\_func

```
io -4 0xfd5f0000 0x000f0001
```

(2)GPIO2\_C3 -> eth0refclk 50/25M refclko25m\_eth0

```
echo 25000000 > /sys/kernel/debug/clk/refclko25m_eth0_out/clk_rate
echo 1 > /sys/kernel/debug/clk/refclko25m_eth0_out/clk_prepare_enable
io -4 0xfd5f8050 0xf00001000
```

(3)GPIO4\_C3 -> gmac0\_clkout 125/50M clk\_gmac\_50m\_cru\_i

```
echo 50000000 > /sys/kernel/debug/clk/clk_gmac_50m/clk_rate
echo 1 > /sys/kernel/debug/clk/clk_gmac_50m/clk_prepare_enable
echo 125000000 > /sys/kernel/debug/clk/clk_gmac_125m/clk_rate
echo 1 > /sys/kernel/debug/clk/clk_gmac_125m/clk_prepare_enable
io -4 0xfd5f8090 0xf0001000
```

(4)GPIO2\_C5 -> 32kout 32K clk\_deepslow



```
io -4 0xfd5f8054 0x0f00010
```

(5)gpio4\_b1 -> clk\_mipi\_camaraout\_m0

```
io -4 0xfd5f8088 0x0f00010
echo 24000000 > /sys/kernel/debug/clk/clk_mipi_camaraout_m0/clk_rate
```

(6)gpio1\_b6 -> clk\_mipi\_camaraout\_m1

```
io -4 0xfd5f802c 0x0f000200
echo 24000000 > /sys/kernel/debug/clk/clk_mipi_camaraout_m1/clk_rate
```

(7)gpio1\_b7 -> clk\_mipi\_camaraout\_m2

```
io -4 0xfd5f802c 0xf0002000
echo 24000000 > /sys/kernel/debug/clk/clk_mipi_camaraout_m2/clk_rate
```

(8)gpio1\_d6 -> clk\_mipi\_camaraout\_m3

```
io -4 0xfd5f803c 0x0f000200
echo 24000000 > /sys/kernel/debug/clk/clk_mipi_camaraout_m3/clk_rate
```

(9)gpio1\_d7 -> clk\_mipi\_camaraout\_m4

```
io -4 0xfd5f803c 0xf0002000
echo 24000000 > /sys/kernel/debug/clk/clk_mipi_camaraout_m4/clk_rate
```

(10)gpio4\_b4 -> clk\_cifout\_out

```
io -4 0xfd5f808c 0x000f0001
echo 24000000 > /sys/kernel/debug/clk/clk_cifout_out/clk_rate
```

(11)gpio3\_a6 -> eth1refclk 50/25M refclko25m\_eth0

```
echo 25000000 > /sys/kernel/debug/clk/refclko25m_eth1_out/clk_rate
echo 1 > /sys/kernel/debug/clk/refclko25m_eth1_out/clk_prepare_enable
io -4 0xfd5f8064 0x0f000100
```

(12)gpio4\_d5 -> testclkoutm0

litcore 10分频

```
io -4 0xfd818314 0x00ff0009
io -4 0xfd7c0328 0x0e000600
io -4 0xfd818808 0x200000
io -4 0xfd7c0808 0x500000
io -4 0xfd5f809c 0x0f00040
```

## 11. RK3528

(1) REF\_CLK\_OUT\_M0 测试点gpio0\_a1

```
io -4 0xFF4B0808 0x00100000
io -4 0xFF540000 0x00f00010
```

(2) REF\_CLK\_OUT\_M2 测试点gpio3\_c3

```
io -4 0xFF4B0808 0x00100000
io -4 0xFF560070 0xf0006000
```

(3) CLK\_32K\_OUT\_M0 测试点gpio3\_c3

```
io -4 0xFF560070 0xf0003000
```

(4) CLK\_32K\_OUT\_M1 测试点gpio1\_c3

```
io -4 0xFF560030 0xf0001000
```

(5) TESTCLK 测试点gpio2\_a5

gpll 10分频输出到testclk

```
io -4 0xFF570044 0x00f00060
io -4 0xFF4A0394 0x03e00120
io -4 0xFF4A0394 0x3c000000
```

备注：如果想测试其他频率或者频率点，按照上面寄存器查找TRM，有详细说明。

## 1.5 GPIO输出CLK软件修改

先确认IO是否支持CLK输出（IOMUX功能可以看到），确认CLK支持哪些频率。如果是TESTCLK功能的，需要查询寄存器，选择不同的功能输出。如果是特定功能的，直接设置频率就可以。

### 1. RK3288

DTS:

```
pinctrl-names = "default";
pinctrl-0 = <&test_clkout>;
assigned-clocks = <&cru SCLK_TESTOUT_SRC>;
assigned-clock-parents = <&xin24m>;
clocks = <&cru SCLK_TESTOUT>;
clock-names = "soc_24M";
```

KERNEL:

```
diff --git a/drivers/clk/rockchip/clk-rk3288.c b/drivers/clk/rockchip/clk-
rk3288.c
index 2784a7ed05db..a45afa6897a7 100644
--- a/drivers/clk/rockchip/clk-rk3288.c
+++ b/drivers/clk/rockchip/clk-rk3288.c
@@ -204,6 +204,12 @@ PNAME(mux_hsadcout_p)      = { "hsadc_src", "ext_hsadc" };
PNAME(mux_edp_24m_p)      = { "ext_edp_24m", "xin24m" };
PNAME(mux_tspout_p)       = { "cpll", "gpll", "nppll", "xin27m" };
```

```

+PNAME(mux_testout_src_p) = { "aclk_peri", "armclk", "aclk_vio0", "ddrphy",
+
+                               "aclk_vcodec", "aclk_gpu", "sclk_rga", "aclk_cpu",
+
+                               "xin24m", "xin27m", "xin32k", "clk_wifi",
+
+                               "dclk_vop0", "dclk_vop1", "sclk_isp_jpe",
+
+                               "sclk_isp" };
+
PNAME(mux_usbphy480m_p)          = { "sclk_otgphy1_480m",
"sclk_otgphy2_480m",
                                "sclk_otgphy0_480m" };
PNAME(mux_hsicphy480m_p)        = { "cpll", "gpll", "usbphy480m_src" };
@@ -560,6 +566,12 @@ static struct rockchip_clk_branch rk3288_clk_branches[]
__initdata = {
                                RK3288_CLKSEL_CON(2), 0, 6, DFLAGS,
                                RK3288_CLKGATE_CON(2), 7, GFLAGS),

+    MUX(SCLK_TESTOUT_SRC, "sclk_testout_src", mux_testout_src_p, 0,
+        RK3288_MISC_CON, 8, 4, MFLAGS),
+    COMPOSITE_NOMUX(SCLK_TESTOUT, "sclk_testout", "sclk_testout_src", 0,
+        RK3288_CLKSEL_CON(2), 8, 5, DFLAGS,
+        RK3288_CLKGATE_CON(4), 15, GFLAGS),
+
    COMPOSITE_NOMUX(SCLK_SARADC, "sclk_saradc", "xin24m", 0,
        RK3288_CLKSEL_CON(24), 8, 8, DFLAGS,
        RK3288_CLKGATE_CON(2), 8, GFLAGS),
diff --git a/include/dt-bindings/clock/rk3288-cru.h b/include/dt-
bindings/clock/rk3288-cru.h
index 1f9c62f07389..61ae793438b4 100644
--- a/include/dt-bindings/clock/rk3288-cru.h
+++ b/include/dt-bindings/clock/rk3288-cru.h
@@ -100,6 +100,8 @@
#define SCLK_MAC_PLL          150
#define SCLK_MAC              151
#define SCLK_MACREF_OUT      152
+#define SCLK_TESTOUT_SRC    153
+#define SCLK_TESTOUT        154

#define DCLK_VOP0             190
#define DCLK_VOP1             191

```

## 2. RK3328

DTS:

```

pinctrl-names = "default";
pinctrl-0 = <&test_clkout>;
assigned-clocks = <&cru SCLK_TESTOUT>;
assigned-clock-parents = <&xin24m>;
assigned-clock-rates = <24000000>;
clocks = <&cru SCLK_TESTOUT>;
clock-names = "soc_24M";

```

KERNEL:

```

diff --git a/drivers/clk/rockchip/clk-rk3328.c b/drivers/clk/rockchip/clk-
rk3328.c
index 1310cfe7798b..5b4ab3360e56 100644

```

```

--- a/drivers/clk/rockchip/clk-rk3328.c
+++ b/drivers/clk/rockchip/clk-rk3328.c
@@ -222,6 +222,14 @@ PNAME(mux_mac2io_ext_p)          = { "clk_mac2io",
                                "gmac_clkin" };
PNAME(mux_i2s_plls_p)          = { "cpll", "dummy_gp1l" };

+PNAME(mux_sclk_testout_p) =    { "clk_wifi", "dummy", "armclk", "sclk_ddrc",
+                                "aclk_rkvdec_pre", "aclk_rkvenc",
+"aclk_vpu_pre",
+                                "aclk_rga_pre", "aclk_vio_pre", "aclk_vop_pre",
+                                "aclk_gpu_pre", "aclk_bus_pre",
+"aclk_peri_pre",
+                                "aclk_gmac", "dclk_lcdc", "clk_pdm",
+"clk_rga",
+                                "sclk_vdec_core", "sclk_venc_core", "clk_tsp",
+                                "dummy", "dummy", "dummy", "xin24m"};
+
static struct rockchip_pll_clock rk3328_pll_clks[] __initdata = {
    [apll] = PLL(pll_rk3328, PLL_APLL, "apll", mux_pll_p,
                0, RK3328_PLL_CON(0),
@@ -836,6 +844,10 @@ static struct rockchip_clk_branch rk3328_clk_branches[]
__initdata = {
    RK3328_SDMMC_EXT_CON0, 1),
    MMC(SCLK_SDMMC_EXT_SAMPLE, "sdmmc_ext_sample", "clk_sdmmc_ext",
        RK3328_SDMMC_EXT_CON1, 1),
+
+    COMPOSITE_DIV_OFFSET(SCLK_TESTOUT, "sclk_testout", mux_sclk_testout_p,
CLK_SET_RATE_NO_REPARENT,
+        RK3328_MISC_CON, 8, 5, MFLAGS, RK3328_CLKSEL_CON(2), 0, 5,
DFLAGS,
+        RK3328_CLKGATE_CON(0), 9, GFLAGS),
};

static const char *const rk3328_critical_clocks[] __initconst = {
diff --git a/include/dt-bindings/clock/rk3328-cru.h b/include/dt-
bindings/clock/rk3328-cru.h
index 62479fddb96b..12f205e31273 100644
--- a/include/dt-bindings/clock/rk3328-cru.h
+++ b/include/dt-bindings/clock/rk3328-cru.h
@@ -98,6 +98,7 @@
#define SCLK_MAC2IO          100
#define SCLK_MAC2PHY        101
#define SCLK_MAC2IO_EXT     102
+#define SCLK_TESTOUT        103

/* dclk gates */
#define DCLK_LCDC            120

```

### 3. PX30

DTS:

```

pinctrl-names = "default";
pinctrl-0 = <&test_clkout>;
assigned-clocks = <&cru SCLK_TEST_OUT>;
assigned-clock-parents = <&xin24m>;
assigned-clock-rates = <24000000>;
clocks = <&cru SCLK_TEST_OUT>;
clock-names = "soc_24M";

```

KERNEL:

```

diff --git a/drivers/clk/rockchip/clk-px30.c b/drivers/clk/rockchip/clk-px30.c
index 9f7a1f91627a..cc4c2fab875d 100644
--- a/drivers/clk/rockchip/clk-px30.c
+++ b/drivers/clk/rockchip/clk-px30.c
@@ -189,6 +189,13 @@ PNAME(mux_wifi_pmu_p) = { "xin24m",
"clk_wifi_pmu_src" };
PNAME(mux_uart0_pmu_p) = { "clk_uart0_pmu_src", "clk_uart0_np5",
"clk_uart0_frac" };
PNAME(mux_usbphy_ref_p) = { "xin24m", "clk_ref24m_pmu" };
PNAME(mux_mipidsiphy_ref_p) = { "xin24m", "clk_ref24m_pmu" };
+PNAME(mux_sclk_test_out_p) = { "armclk", "aclk_gpu", "clk_ddrphy4x",
"clk_i2c0", "aclk_vo_pre",
+ "aclk_rga", "dclk_vopb", "dclk_vopl",
"aclk_vpu_pre", "aclk_vi_pre",
+ "clk_isp", "clk_rtc32k_frac", "clk_ddrphy1x",
"aclk_peri_pre",
+ "dummy", "dummy", "dummy", "dummy",
"clk_pwm0",
+ "dummy", "aclk_crypto_pre", "clk_crypto_apk",
"xin24m", "aclk_gmac_pre",
+ "clk_gmac", "aclk_bus_pre", "clk_pdm",
"clk_i2s0_tx_out", "clk_tsadc",
+ "clk_uart1", "clk_saradc", "clk_otp"};

static struct rockchip_pll_clock px30_pll_clks[] __initdata = {
[apll] = PLL(pll_rk3328, PLL_APLL, "apll", mux_pll_p,
@@ -907,6 +914,11 @@ static struct rockchip_clk_branch px30_clk_branches[]
__initdata = {
PX30_CLKGATE_CON(8), 1, GFLAGS),
GATE(PCLK_GMAC, "pclk_gmac", "pclk_gmac_pre", 0,
PX30_CLKGATE_CON(8), 3, GFLAGS),
+
+ COMPOSITE(SCLK_TEST_OUT, "sclk_test_out", mux_sclk_test_out_p,
CLK_SET_RATE_NO_REPARENT,
+ PX30_CLKSEL_CON(57), 0, 5, MFLAGS, 8, 4, DFLAGS,
+ PX30_CLKGATE_CON(16), 15, GFLAGS),
+
};

static struct rockchip_clk_branch px30_gpu_src_clk[] __initdata = {
diff --git a/include/dt-bindings/clock/px30-cru.h b/include/dt-
bindings/clock/px30-cru.h
index 644d1f5d26d0..648d7b5ae3a5 100644
--- a/include/dt-bindings/clock/px30-cru.h
+++ b/include/dt-bindings/clock/px30-cru.h
@@ -102,6 +102,7 @@

```

```

#define SCLK_SDMMC_DIV50      87
#define SCLK_I2S0_TX_MUX      88
#define SCLK_I2S0_RX_MUX      89
+#define SCLK_TEST_OUT        90

```

#### 4. RK3308

DTS:

```

pinctrl-names = "default";
pinctrl-0 = <&test_clkout>;
assigned-clocks = <&cru SCLK_TESTOUT>;
assigned-clock-parents = <&xin24m>;
assigned-clock-rates = <24000000>;
clocks = <&cru SCLK_TESTOUT>;
clock-names = "soc_24M";

```

KERNEL:

```

diff --git a/drivers/clk/rockchip/clk-rk3308.c b/drivers/clk/rockchip/clk-
rk3308.c
index 9d09c516bea5..fba208e1a09f 100644
--- a/drivers/clk/rockchip/clk-rk3308.c
+++ b/drivers/clk/rockchip/clk-rk3308.c
@@ -192,6 +192,14 @@ PNAME(mux_spdif_tx_p)          = { "clk_spdif_tx_src",
"clk_spdif_tx_frac", "mclk_i2s0_2
PNAME(mux_spdif_rx_src_p)      = { "clk_spdif_rx_div", "clk_spdif_rx_div50" };
PNAME(mux_spdif_rx_p)          = { "clk_spdif_rx_src", "clk_spdif_rx_frac" };
PNAME(mux_uart_src_p)          = { "xin24m", "usb480m", "dpll", "vp1l0", "vp1l1"
};
+PNAME(mux_sclk_testout_p) =    { "xin24m", "armclk", "dummy", "dummy",
+
+                                "aclk_peri", "hclk_peri", "clk_nandc",
+                                "clk_sdmmc", "clk_sdio", "clk_emmc",
+                                "clk_sfc", "dummy", "aclk_bus",
+                                "hclk_bus", "clk_crypto", "clk_crypto_apk",
+"dclk_vop",
+
+                                "clk_uart0", "clk_i2c0", "clk_spi0",
+                                "clk_tsadc", "clk_saradc", "clk_otp",
+"hclk_audio"};
+
static u32 uart_src_mux_idx[] = { 4, 3, 0, 1, 2 };

static struct rockchip_pll_clock rk3308_pll_clks[] __initdata = {
@@ -899,6 +907,10 @@ static struct rockchip_clk_branch rk3308_clk_branches[]
__initdata = {
        GATE(PCLK_PWM2, "pclk_pwm2", "pclk_bus", CLK_IGNORE_UNUSED,
RK3308_CLKGATE_CON(7), 13, GFLAGS),
        GATE(PCLK_CAN, "pclk_can", "pclk_bus", CLK_IGNORE_UNUSED,
RK3308_CLKGATE_CON(7), 14, GFLAGS),
        GATE(PCLK_OWIRE, "pclk_owire", "pclk_bus", CLK_IGNORE_UNUSED,
RK3308_CLKGATE_CON(7), 15, GFLAGS),
+
+        COMPOSITE(SCLK_TESTOUT, "sclk_testout", mux_sclk_testout_p,
CLK_SET_RATE_NO_REPARENT,
+
+                RK3308_CLKSEL_CON(73), 8, 5, MFLAGS, 0, 5, DFLAGS,
+                RK3308_CLKGATE_CON(4), 9, GFLAGS),

```

```
};

static struct rockchip_clk_branch rk3308_clk_ddrphy[] __initdata = {
diff --git a/include/dt-bindings/clock/rk3308-cru.h b/include/dt-
bindings/clock/rk3308-cru.h
index 5088a0f6fb02..c4707e8775b6 100644
--- a/include/dt-bindings/clock/rk3308-cru.h
+++ b/include/dt-bindings/clock/rk3308-cru.h
@@ -133,6 +133,7 @@
#define SCLK_PWM1          120
#define SCLK_PWM2          121
#define SCLK_OWIRE         122
+define SCLK_TESTOUT      123

/* dclk */
#define DCLK_VOP            125
```

## 5. RV1108

DTS:

```
pinctrl-names = "default";
pinctrl-0 = <&test_clkout>;
assigned-clocks = <&cru SCLK_TESTOUT>;
assigned-clock-parents = <&xin24m>;
assigned-clock-rates = <24000000>;
clocks = <&cru SCLK_TESTOUT>;
clock-names = "soc_24M";
```

KERNEL:

```
--- a/drivers/clk/rockchip/clk-rv1108.c
+++ b/drivers/clk/rockchip/clk-rv1108.c
@@ -159,6 +159,11 @@ PNAME(mux_dclk_hdmiphy_p) = { "hdmiphy", "xin24m" };
PNAME(mux_dclk_vop_p) = { "dclk_hdmiphy", "dclk_vop_src" };
PNAME(mux_hdmi_cec_src_p) = { "dpll", "gpll", "xin24m" };
PNAME(mux_cvbs_src_p) = { "apll", "io_cvbs_clkin", "hdmiphy", "gpll" };
+PNAME(mux_sclk_testout_p) = { "armclk", "aclk_bus_pre", "aclk_vio0",
+ "aclk_vio1",
+ "aclk_periph", "sclk_dsp", "aclk_rkvdec",
+ "aclk_rkvenc", "xin24m", "dummy",
+ "dclk_vop", "sclk_wifi", "sclk_rga",
+ "sclk_isp", "aclk_vpu", "clk_venc_core"};

static struct rockchip_pll_clock rv1108_pll_clks[] __initdata = {
    [apll] = PLL(pll_rk3399, PLL_APLL, "apll", mux_pll_p, 0,
RV1108_PLL_CON(0),
@@ -776,6 +781,10 @@ static struct rockchip_clk_branch rv1108_clk_branches[]
__initdata = {

    MMC(SCLK_EMMC_DRV, "emmc_drv", "sclk_emmc", RV1108_EMMC_CON0,
1),
    MMC(SCLK_EMMC_SAMPLE, "emmc_sample", "sclk_emmc", RV1108_EMMC_CON1,
1),
+
+    COMPOSITE_DIV_OFFSET(SCLK_TESTOUT, "sclk_testout", mux_sclk_testout_p,
CLK_SET_RATE_NO_REPARENT,
```

```

+         RV1108_MISC_CON, 8, 4, MFLAGS, RV1108_CLKSEL_CON(39), 8, 5,
DFLAGS,
+         RV1108_CLKGATE_CON(9), 9, GFLAGS),
};

```

```

static const char *const rv1108_critical_clocks[] __initconst = {
diff --git a/include/dt-bindings/clock/rv1108-cru.h b/include/dt-
bindings/clock/rv1108-cru.h
index d8d0e0456dc2..b1cbd24a78eb 100644
--- a/include/dt-bindings/clock/rv1108-cru.h
+++ b/include/dt-bindings/clock/rv1108-cru.h
@@ -86,6 +86,7 @@
#define SCLK_UART0_SRC            127
#define SCLK_UART1_SRC            128
#define SCLK_UART2_SRC            129
+#define SCLK_TESTOUT              130

#define DCLK_VOP_SRC              185
#define DCLK_HDMIPHY              186

```

## 6. RV1126

DTS:

```

pinctrl-names = "default";
pinctrl-0 = <&test_clkout>;
assigned-clocks = <&cru SCLK_TESTOUT>;
assigned-clock-parents = <&xin24m>;
assigned-clock-rates = <24000000>;
clocks = <&cru SCLK_TESTOUT>;
clock-names = "soc_24M";

```

KERNEL:

```

diff --git a/drivers/clk/rockchip/clk-rv1126.c b/drivers/clk/rockchip/clk-
rv1126.c
index 05dea9fbd1f0..39905572ca13 100644
--- a/drivers/clk/rockchip/clk-rv1126.c
+++ b/drivers/clk/rockchip/clk-rv1126.c
@@ -220,6 +220,14 @@ PNAME(mux_gppll_cppll_hppll_p) = { "gppll",
"dummy_cppll", "dummy_hppll" };
PNAME(mux_gppll_cppll_apll_hppll_p) = { "gppll", "dummy_cppll", "dummy_apll",
"dummy_hppll" };
#endif

+PNAME(mux_sclk_testout_p) = { "xin24m", "clk_rtc32k", "armclk", "dummy",
+
+ "aclk_pdbus", "hclk_pdbus", "clk_core_npu",
+ "aclk_pdnpu", "aclk_pdvdec",
+ "clk_vdec_hevc_ca",
+
+ "aclk_pdvepu", "clk_venc_core", "aclk_pdispp",
+ "aclk_pdjpeg", "aclk_pdvi", "aclk_pdvo",
+ "dclk_vop",
+
+ "clk_spi1", "dummy", "sclk_uart3",
+ "aclk_pdphp", "sclk_sfc", "dummy", "dummy"};
+
static u32 rgmii_mux_idx[] = { 2, 3, 0, 1 };

```



```

static struct rockchip_pll_clock rv1126_pmu_pll_clks[] __initdata = {
@@ -1378,6 +1386,10 @@ static struct rockchip_clk_branch rv1126_clk_branches[]
__initdata = {
    GATE(PCLK_DDRPHY, "pclk_ddrphy", "pclk_pdttop", CLK_IGNORE_UNUSED,
        RV1126_CLKGATE_CON(23), 0, GFLAGS),
    #endif
+
+    COMPOSITE(SCLK_TESTOUT, "sclk_testout", mux_sclk_testout_p,
CLK_SET_RATE_NO_REPARENT,
+        RV1126_CLKSEL_CON(75), 8, 5, MFLAGS, 0, 5, DFLAGS,
+        RV1126_CLKGATE_CON(23), 13, GFLAGS),
    };

static const char *const rv1126_cru_critical_clocks[] __initconst = {
diff --git a/include/dt-bindings/clock/rv1126-cru.h b/include/dt-
bindings/clock/rv1126-cru.h
index 474bcbc546af..2f18e3127113 100644
--- a/include/dt-bindings/clock/rv1126-cru.h
+++ b/include/dt-bindings/clock/rv1126-cru.h
@@ -213,6 +213,7 @@
#define CLK_NPUPVTM            146
#define SCLK_DDRCLK            147
#define CLK_OTP                148
+#define SCLK_TESTOUT          149

/* dclk */
#define DCLK_DECOM              150

```

## 7. RK3368

DTS:

```

pinctrl-names = "default";
pinctrl-0 = <&test_clkout>;
assigned-clocks = <&cru SCLK_TESTOUT>;
assigned-clock-parents = <&xin24m>;
assigned-clock-rates = <24000000>;
clocks = <&cru SCLK_TESTOUT>;
clock-names = "soc_24M";

```

KERNEL:

```

diff --git a/drivers/clock/rockchip/clock-rk3368.c b/drivers/clock/rockchip/clock-
rk3368.c
index 009abde551aa..d5a7f720ceb2 100644
--- a/drivers/clock/rockchip/clock-rk3368.c
+++ b/drivers/clock/rockchip/clock-rk3368.c
@@ -157,6 +157,11 @@ PNAME(mux_uart3_p) = { "uart3_src", "uart3_frac",
"xin24m" };
PNAME(mux_uart4_p) = { "uart4_src", "uart4_frac", "xin24m" };
PNAME(mux_mac_p) = { "mac_pll_src", "ext_gmac" };
PNAME(mux_mmc_src_p) = { "cpll", "gppll", "usbphy_480m", "xin24m" };
+PNAME(mux_sclk_testout_p) = { "aclk_peri", "armclk", "aclk_vio0", "dummy",
+
+    "aclk_video", "sclk_gpu_core", "dummy",
+
+    "dummy", "xin24m", "aclk_cci_pre",
+
+    "xin32k", "dummy", "dclk_vop",
+
+    "armclk1", "aclk_gpu_src", "sclk_isp"};

```

```

static struct rockchip_pll_clock rk3368_pll_clks[] __initdata = {
    [apllb] = PLL(pll_rk3066, PLL_APLLb, "apllb", mux_pll_p, 0,
RK3368_PLL_CON(0),
@@ -881,6 +886,10 @@ static struct rockchip_clk_branch rk3368_clk_branches[]
__initdata = {
    GATE(SCLK_TIMER02, "sclk_timer02", "xin24m", CLK_IGNORE_UNUSED,
RK3368_CLKGATE_CON(24), 2, GFLAGS),
    GATE(SCLK_TIMER01, "sclk_timer01", "xin24m", CLK_IGNORE_UNUSED,
RK3368_CLKGATE_CON(24), 1, GFLAGS),
    GATE(SCLK_TIMER00, "sclk_timer00", "xin24m", CLK_IGNORE_UNUSED,
RK3368_CLKGATE_CON(24), 0, GFLAGS),
+
+    COMPOSITE_DIV_OFFSET(SCLK_TESTOUT, "sclk_testout", mux_sclk_testout_p,
CLK_SET_RATE_NO_REPARENT,
+    RK3368_MISC_CON, 0, 4, MFLAGS, RK3368_CLKSEL_CON(47), 8, 5,
DFLAGS,
+    RK3368_CLKGATE_CON(7), 1, GFLAGS),
};

static const char *const rk3368_critical_clocks[] __initconst = {
diff --git a/include/dt-bindings/clock/rk3368-cru.h b/include/dt-
bindings/clock/rk3368-cru.h
index 5d3531686790..c6e36b596d24 100644
--- a/include/dt-bindings/clock/rk3368-cru.h
+++ b/include/dt-bindings/clock/rk3368-cru.h
@@ -94,6 +94,7 @@
#define SCLK_DDRCLK            139
#define SCLK_TSP              140
#define SCLK_HSADC_TSP        141
+define SCLK_TESTOUT         142

#define DCLK_VOP              190
#define MCLK_CRYPT0          191

```

## 8. RK3399

DTS:

```

pinctrl-names = "default";
pinctrl-0 = <&test_clkout>;
assigned-clocks = <&cru SCLK_TESTCLKOUT1>;
assigned-clock-rates = <24000000>;
clocks = <&cru SCLK_TESTCLKOUT1>;
clock-names = "soc_24M";

```

```

pinctrl-names = "default";
pinctrl-0 = <&test_clkout>;
assigned-clocks = <&cru SCLK_TESTCLKOUT2>;
assigned-clock-rates = <24000000>;
clocks = <&cru SCLK_TESTCLKOUT2>;
clock-names = "soc_24M";

```

## 9. RK3566/8

DTS:

```
pinctrl-names = "default";
pinctrl-0 = <&test_clkout>;
assigned-clocks = <&pmucru CLK_WIFI>;
assigned-clock-rates = <24000000>;
clocks = <&pmucru CLK_WIFI>;
clock-names = "soc_24M";
```

#### 10. RK3588

DTS:

```
pinctrl-names = "default";
pinctrl-0 = <&test_clkout>;
assigned-clocks = <&cru CLK_CIFOUT_OUT>;
assigned-clock-rates = <24000000>;
clocks = <&cru CLK_CIFOUT_OUT>;
clock-names = "soc_24M";
```

#### 11. RK3528

- CLK\_TESTOUT: 暂不开发对外使用，用于内部自测信号使用。
- CLK\_REFOUT: 时钟频率默认只有24M，可开关，不支持频率修改。
- CLK\_DEEPSLOW: 时钟频率默认32K，不支持开关和频率修改。

所以只需配置Pinctrl即可。

备注：

dts中pinctrl的控制，如果疑问参考pinctrl的文档。

RK3568\RK3588 IO比较丰富，如果想要其他的IO输出CLK，请先查看IO和CLK的对应关系，然后按照示例操作。