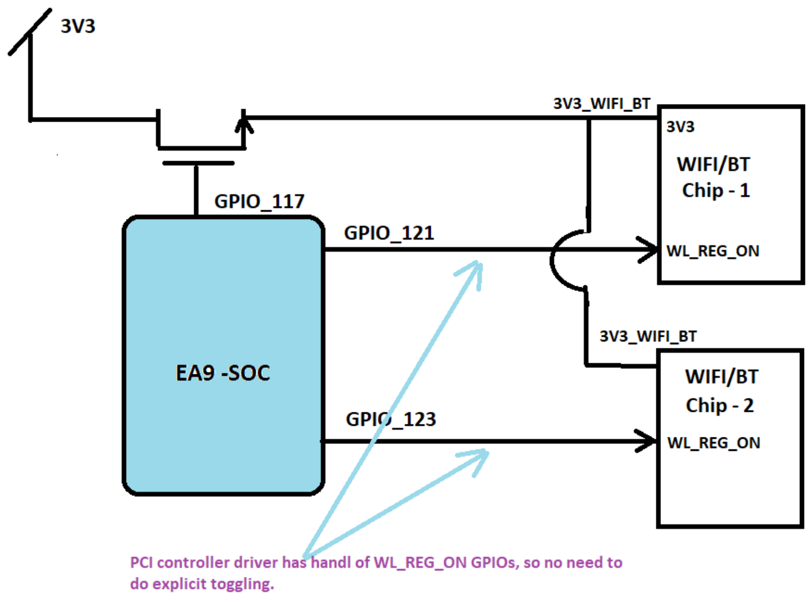
# WiFi-BT chip Power Enable and Suspend

| Version | State | Date | Remark |
|---|---|---|---|
| 1.0 | Draft | 20 Feb 2023 | In progress |

## Power ON GPIO

In B2 hardware Power to WiFi-BT hardware is made switchable using a MOS switch and a GPIO pin of SOC. It is explained as below.



In Normal/default mode, WiFi_BT_Power_On_GPIO is being set to Output - High, so that WiFi-BT chip can be turned ON. Then after only subsequent operation of Enumeration is triggred by below command.

```
# echo 1 > /sys/devices/platform/161c0000.pcie/pcie_sysfs      (For PCIe channel - 2)
# echo 1 > /sys/devices/platform/161c1000.pcie/pcie_sysfs      (For PCIe channel - 3)
```

As, WiFi-BT chip is a power hungry device, so for power management/saving it must be turned off in power down mode.

For this, pinctrl related DTS changes are done as below for WiFi_BT_Power_On_GPIO or GPP1_1 or GPIO_117.

**Method - 1**

```
&pinctrl_peric0 {
        .......
        .......
        wifibt_power_enable_pin_active: wifibt_power_enable_pin_active {
                samsung,pins = "gpp1-1";
                samsung,pin-function = <EXYNOS_PIN_FUNC_OUTPUT>;
                samsung,pin-pud = <EXYNOS_PIN_PULL_UP>;
                samsung,pin-drv = <EXYNOS5420_PIN_DRV_LV1>;
                samsung,pin-val = <1>;
        };

        wifibt_power_enable_pin_sleep: wifibt_power_enable_pin_sleep {
                samsung,pins = "gpp1-1";
                samsung,pin-function = <EXYNOS_PIN_FUNC_OUTPUT>;
                samsung,pin-pud = <EXYNOS_PIN_PULL_DOWN>;
                samsung,pin-drv = <EXYNOS5420_PIN_DRV_LV1>;
                samsung,pin-val = <0>;
        };
};

&pcie_2 {
        status = "okay";
        use-msi = "true";
        use-bifurcation = "true";
        num-lanes = <1>;
        pinctrl-names = "default", "sleep";

        pinctrl-0 = <&pcie_clkreq2 &pcie_perst2_out &wifi_reg_ch2 &wifibt_power_enable_pin_active>;
        pinctrl-1 = <&pcie_clkreq2 &pcie_perst2_out &wifi_reg_ch2 &wifibt_power_enable_pin_sleep>;
        gpio_wifi_reg_on = <&gpp1 5 GPIO_ACTIVE_LOW>;
};
```

## Pinctrl state "default" and "idle"

By doing above dts change by using pinctrl states having pinctrl-names "default" and "sleep" did not work, that is by executing Suspend command Power-On GPIO was not turning to LOW level to disable power.

Then by doing code walk-through of PCIe-dwc driver it was found that it defines two pinctrl states "default" and "idle" instead of using "defualt" and "sleep".

```
exynos_v920_pcie_probe()
{
         .....
         exynos_v920_pcie_pinconfig()
         {
                 .....
          }

}

exynos_v920_pcie_pinconfig()
{
        .........
        exynos_pcie->pcie_pinctrl = devm_pinctrl_get(pci->dev);

        exynos_pcie->pin_state[PCIE_PIN_DEFAULT] =
                pinctrl_lookup_state(exynos_pcie->pcie_pinctrl, "default");

        exynos_pcie->pin_state[PCIE_PIN_IDLE] =
                pinctrl_lookup_state(exynos_pcie->pcie_pinctrl, "idle");

        .......
}
```

## Use of Pinctrl-state "idle" at Suspend

By doing code walk through it is found that at the time of suspend the PCIe controller driver's suspend callback "exynos_v920_pcie_suspend_noirq() " is

invoked, which inturn calls the function "exynos_v920_pcie_poweroff()"  and it eventually calls the function "pinctrl_select_state()" with PINCTRL_STATE_IDLE as argument.

Code snippet is as below.

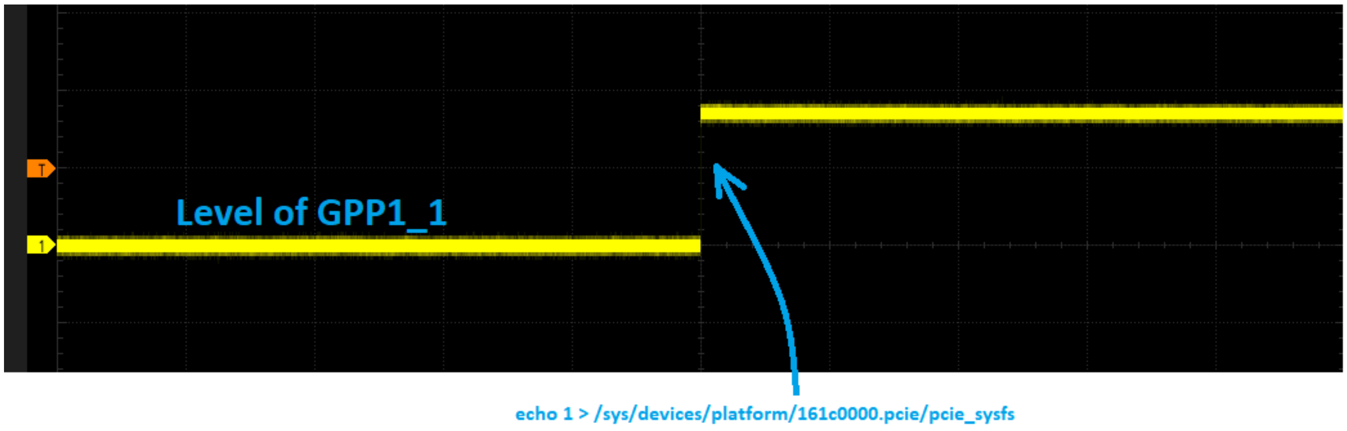
```
exynos_v920_pcie_suspend_noirq()
{
        .......
        exynos_v920_pcie_poweroff()
        {
                ........
                pinctrl_select_state(exynos_pcie->pcie_pinctrl,
                                exynos_pcie->pin_state[PCIE_PIN_IDLE]);
                ......
        }
}
```

## WiFi_BT_Power_On_GPIO state after bootup and at the time of enumeration

Level of Power-On GPIO is LOW by default after boot-up of the target. Its level changes to HIGH only when "echo 1" command is run to enumerate PCIe devvice(s). So,

Wifi chip does not draw current untill PCIe enumeration is triggered. As soon PCIe enumeration command  - "echo 1 > /sys/devices/platform/161c0000.pcie /pcie_sysfs" is

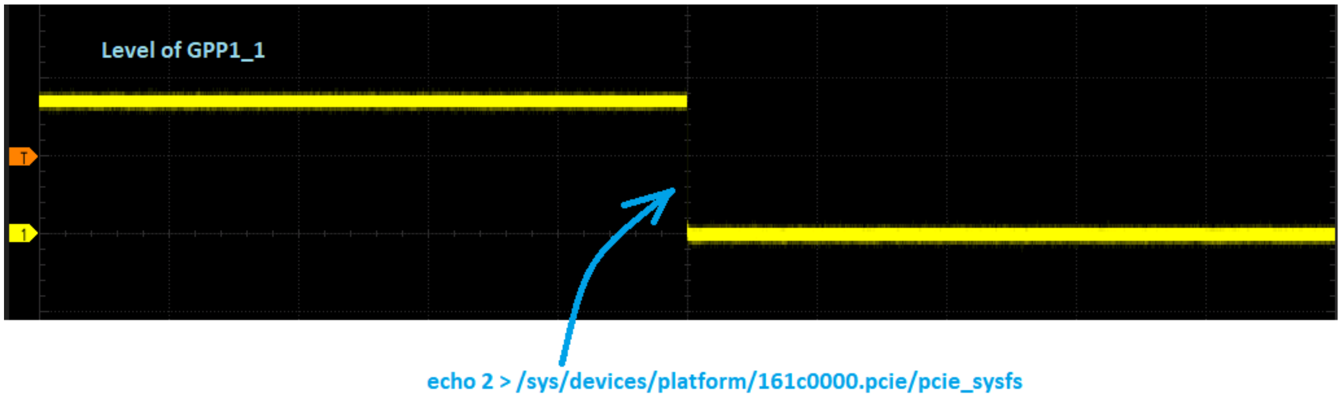
triggered, WiFi_BT_Power_On_GPIO goes HIGH. It is shown in below waveform.



There is one more way to power OFF the WiFi chip, by using below command.

```
# echo 2 > /sys/devices/platform/161c0000.pcie/pcie_sysfs
```

Waveform:



## Verifying Pinctrl settings for WiFi_BT_Power_On_GPIO

```
# cd /sys/kernel/debug/pinctrl/10830000.pinctrl-samsung-pinctrl
# cat pins
O/p -

      .......

      pin 117 (gpp1-1) 1:gpp1  CON(0x1) DAT(0x1) PUD(0x3) DRV(0x0) CON_PDN(0x2) PUD_PDN(0x1)

      .......

       Active mode: -
       --------------------
       CON(0x1) - > Output pin
       DAT(0x1) -> Output Data = 1
       PUD(0x3) -> Pull Up

       Power down mode: -
       -----------------------------
       CON_PDN(0x2) -> Input pin in power down
       PUD_PDN(0x1) -> Pull Down in power down
```

## Procedure of Handling Suspend-To-Ram

```
I. Trigger suspend of IVI (On SYS console):

   # echo -ne '\x01' > /sys/nk/prop/nk.vm.3.s2r

   Switch to IVI console (ESC+O+6+R) and then see if  suspended in 30s and if not then run below command.

   (On IVI console)
   # echo -ne mem > /sys/power/state

   [ If IVI console suspends, then  at this state we can't enter any command.]


II. Trigger the Resume of IVI Partition after some time (On SYS console) :

   # echo -ne '\x00' > /sys/nk/prop/nk.vm.3.s2r
```

## Verification of Suspend-To-Ram

WiFi_BT_Power_On_GPIO i.e. GPP1_1 can be probed and also prints were given in PCIe controller driver's power-management callback functions i.e. exynos_v920_pcie_suspend_noirq() and exynos_v920_pcie_resume_noirq().

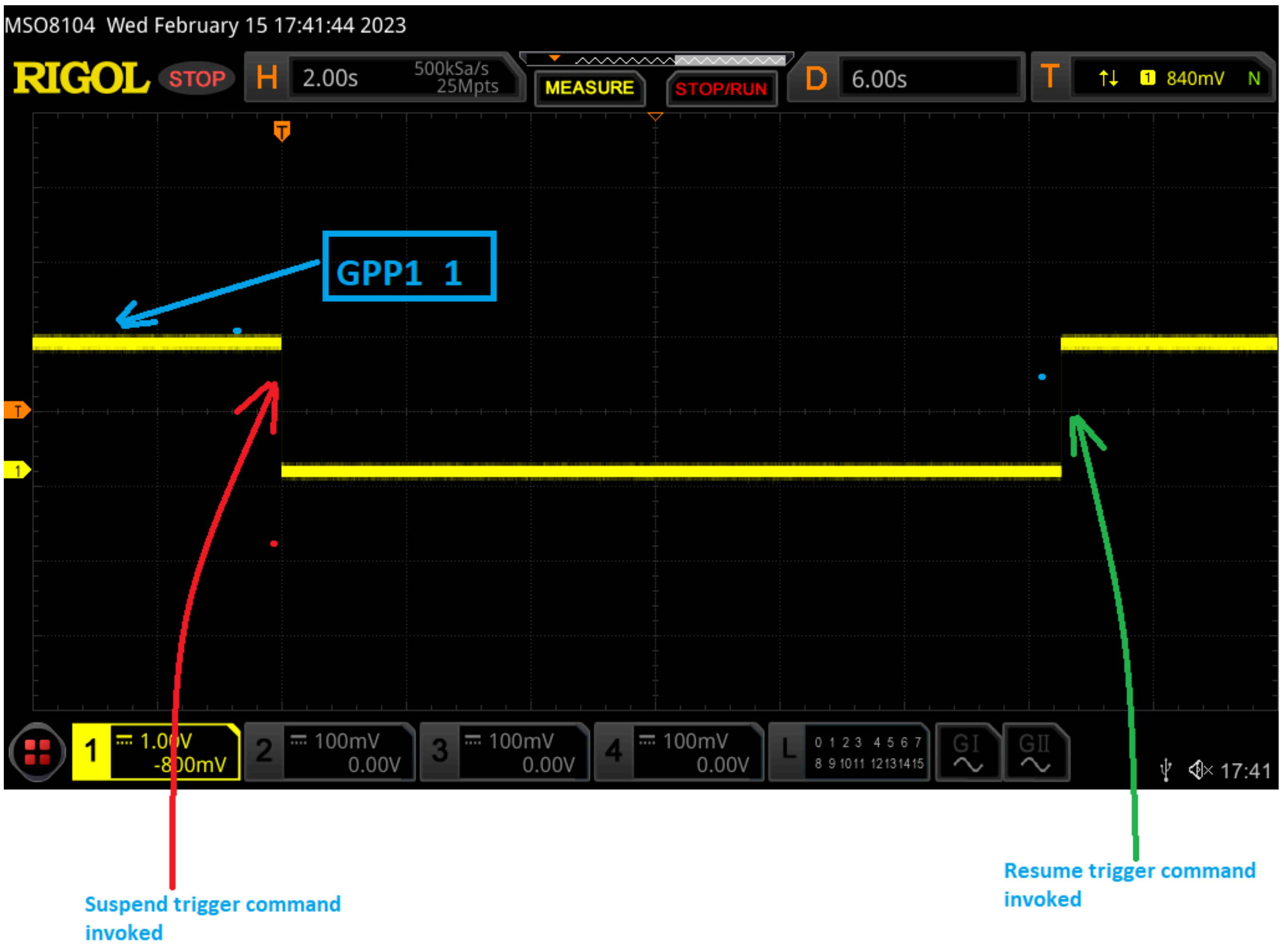
By following Method-1 and Method-2 as shown above, GPP1_1 is outputting High after PCIe controller driver's probe is invoked, but in power down mode, it is not outputting LOW. As on Date - 15-02-2023, it is found that S-2-R is

not properly implemented and so above DTS changes are subject to analysis while stabilizing S-2-R.

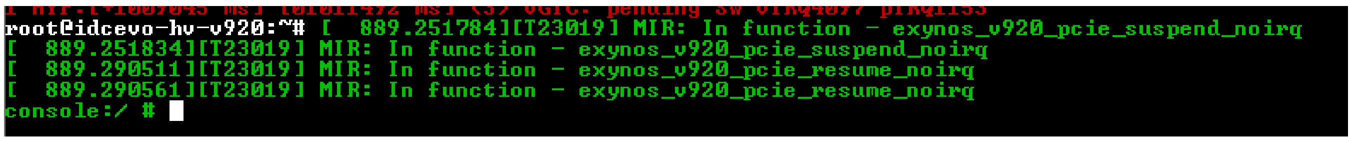
At this moment, pm callbacks of PCIe controller driver is being invoked, confirmed by putting prints. Then by calling gpio_set_value() API, the GPIO level is being set at suspend and resume case, as below.

```
static int exynos_v920_pcie_suspend_noirq(struct device *dev)
{
   ....
   pr_alert("MIR: In function - %s\n", __func__);
   gpio_set_value(117, 0);

   return 0;
}

static int exynos_v920_pcie_resume_noirq(struct device *dev)
{
   ....
   pr_alert("MIR: In function - %s\n", __func__);
   gpio_set_value(117, 1);

   return 0;
}
```

By putting GPIO kernel APIs in PM callbacks of PCIe controller driver, the GPIO - WiFi_BT_Power_On_GPIO changes its state, as depicted in below waveform.


MSO8104 Wed February 15 17:41:44 2023
RIGOL
STOP
H 2.00s
500kSa/s
25Mpts
MEASURE
STOP/RUN
D 6.00s
T
840mV
N
GPP1 1
1 1.00V -800mV
2 100mV 0.00V
3 100mV 0.00V
4 100mV 0.00V
L 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
GI
GII
17:41
Suspend trigger command invoked
Resume trigger command invoked

Dmesg Log: Execution of PCIe controller driver's Suspend Resume callbacks.



```
root@idcevo-hv-v920:~# [  889.251784][T23019] MIR: In function - exynos_v920_pcie_suspend_noirq
[  889.251834][T23019] MIR: In function - exynos_v920_pcie_suspend_noirq
[  889.290511][T23019] MIR: In function - exynos_v920_pcie_resume_noirq
[  889.290561][T23019] MIR: In function - exynos_v920_pcie_resume_noirq
console:/ #
```

## Test Case: Enumeration should succeed after Resuming from Suspend

**Step I. Perform PCIe enumeration in normal mode (after Device Boots )**

```
# echo 1 > /sys/devices/platform/161c0000.pcie/pcie_sysfs
# echo 1 > /sys/devices/platform/161c1000.pcie/pcie_sysfs
```

**Step II. Suspend IVI**

**On SYS console:**

```
# echo -ne '\x01' > /sys/nk/prop/nk.vm.3.s2r
```

**Switch to IVI console (ESC+O+6+R) and then see if  suspended in 30s and if not then run below command.**

**On IVI console:**

```
# echo -ne mem > /sys/power/state
```

**Step III. Resume IVI**

**On SYS console**

```
# echo -ne '\x00' > /sys/nk/prop/nk.vm.3.s2r
```

**Step IV. Peform PCIe enumeration**

```
# echo 1 > /sys/devices/platform/161c0000.pcie/pcie_sysfs
# echo 1 > /sys/devices/platform/161c1000.pcie/pcie_sysfs
```

**Step V. Verify using "lspci" command.**