Rockchip GMAC RGMII Delayline Guide

ID: RK-KF-YF-013

Release Version: V1.2.0

Release Date: 2021-12-28

Security Level: □Top-Secret □Secret □Internal ■Public

Rockchip Electronics Co., Ltd.

No.18 Building, A District, No.89, software Boulevard Fuzhou, Fujian,PRC

Website: www.rock-chips.com

Customer service Tel: +86-4007-700-590

Customer service Fax: +86-591-83951833

Customer service e-Mail: fae@rock-chips.com

Preface

Overview

The product of Rockchip has the function of Gigabit Ethernet and uses the RGMII interface. In order to be compatible with the signal differences brought by various hardware, the chip adds the function adjustment (TX / RX) RGMII delayline. This document describes how to get a suitable set of delayline to achieve the best performance of Gigabit Ethernet, and how to improve the hardware to get the maximum delayline window.

Product Version

Chipset	Kernel Version
ALL chipset	All version

Intended Audience

This document (this guide) is mainly intended for:

Technical support engineers

Software development engineers

Revision History

Version	Author	Date	Change Description
V1.0.0	David Wu	2020-02-07	Initial version
V1.1.0	David Wu	2020-06-24	Support all kernel version
V1.2.0	David Wu	2021-12-28	Support rgmii-rxid mode

Contents

1. How To Get RGMII Delayline

If your project features a Gigabit Ethernet and uses the RGMII interface, as long as there is a hardware difference, you need to reset the delayline configuration. Because if the configured delayline value does not match the hardware of your project, it will affect your Gigabit Ethernet performance and even normal network functions.

1.1 Checking Code

The implementation code is all in the file `drivers/net/ethernet/stmicro/stmmac/dwmac-rk-tool.c`, so it is also more convenient to transplant. If your project does not have this part of the code, please request a patch on Redmine, which has kernel-4.4 and kernel-3.10 versions, the others after kernel-4.4 should have supported it.

- Kernel-4.4 patch: Rockchip_RGMII_Delayline_Kernel4.4.tar.gz
  The kernel-4.4 performance has been optimized, the patch code is generated based on the current code. If there is a problem that it cannot be compiled, please try to patch `kernel4.4_stmmac_optimize_output_performances_20191119.zip` at first.
- Kernel-3.10 patch: Rockchip_RGMII_Delayline_Kernel3.10.tar.gz

1.2 Checking Node

After the code in the previous step is confirmed and compiled, it will generate several sysfs nodes. If it is not generated, it means there is a problem with the patch. Taking RK3399 as an example, you can see these nodes in the directory `/sys/devices/platform/fe300000.ethernet` :

```
rk3399_all:/ # ls -l /sys/devices/platform/fe300000.ethernet/
total 0
lrwxrwxrwx 1 root root    0 2013-01-18 20:21 driver -> ../../../bus/platform/drivers/rk_gmac-dwmac
-rw-r--r-- 1 root root 4096 2013-01-18 20:21 driver_override
--w------- 1 root root 4096 2013-01-18 20:21 mac_lb
drwxr-xr-x 3 root root    0 2013-01-18 08:50 mdio_bus
-r--r--r-- 1 root root 4096 2013-01-18 20:21 modalias
drwxr-xr-x 3 root root    0 2013-01-18 08:50 net
lrwxrwxrwx 1 root root    0 2013-01-18 20:21 of_node -> ../../../firmware/devicetree/base/ethernet@fe300000
--w------- 1 root root 4096 2013-01-18 20:21 phy_lb
--w------- 1 root root 4096 2013-01-18 20:21 phy_lb_scan
drwxr-xr-x 2 root root    0 2013-01-18 08:50 power
-rw-r--r-- 1 root root 4096 2013-01-18 20:21 rgmii_delayline
lrwxrwxrwx 1 root root    0 2013-01-18 20:21 subsystem -> ../../../bus/platform
-rw-r--r-- 1 root root 4096 2013-01-18 08:50 uevent
```

1.3 Usage

Note that if you are using `RTL8211E phy`, you need to remove the network cable before testing.

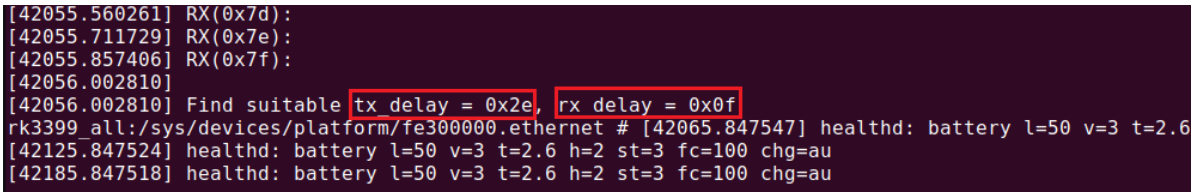
1.3.1 Scanning Delayline Window

Scanning through the `phy_lb_scan` node will get an window, then get the coordinates in the center of this window, which needs to be scanned with a Gigabit speed of 1000.

```
echo 1000 > phy_lb_scan
```

The horizontal axis represents the delayline in the TX direction (coordinate range <0x00, 0x7f>), and the vertical axis represents the delayline in the RX direction (coordinate range is also <0x00, 0x7f>). The "O" indicates that the coordinates of the point can be Okay, and all the other blank spaces indicate loopback failed. Taking RK3399 as an example, through the Gigabit scan command, discard rows or columns with blank spaces in between, you can get a rectangular window, and get the coordinates of the center point. The RX coordinates of the vertical axis have been printed, and the horizontal axis coordinates are not printing due to screen width. It is not displayed, you need to find it manually, starting from RX(0xXX): of:.

And the center point of coordinate will be printed after the scanning window:



The hardware signal of the RK3399 board tested in this picture is not very good, so the window is not very large. Similarly, RGMII 100M can also get a window. `Echo 100 > phy_lb_scan` can see that the 100M window is very large, occupying almost all the coordinates, because 100M is not as demanding as Gigabit on the signal.

1.3.2 Testing Scanned Result

Secondly, configure the scanned value to the `rgmii_delayline` node through the command, and then test whether the TX/RX data transmission under this configuration is normal through the `phy_lb` node test, at least this test needs pass first.

```
echo (tx delayline) (rx delayline) > rgmii_delayline
cat rgmii_delayline
echo 1000 > phy_lb
```

```
rk3399_all:/sys/devices/platform/fe300000.ethernet # echo 0x2e 0x0f > rgmii_de>
[42265.031754] Set rgmii delayline tx: 0x2e, rx: 0xf
rk3399_all:/sys/devices/platform/fe300000.ethernet # cat rgmii_delayline
tx delayline: 0x2e, rx delayline: 0xf
rk3399_all:/sys/devices/platform/fe300000.ethernet # echo 1000 > phy_lb
[42270.042819] PHY loopback speed set to 1000
[42270.738739] PHY loopback: PASS
```

After testing pass, fill the delayline into dts respectively: `tx_delay = <0x2e>;` and `rx_delay = <0x0f>;`, re-burn the firmware, and then continue to test the ping or iperf performance test, generally test to this step will be enough.

```
&gmac {
        phy-supply = <&vcc_lan>;
        clock_in_out = "output";
        assigned-clocks = <&cru SCLK_MAC>, <&cru SCLK_RMII_SRC>;
        assigned-clock-parents = <&cru PLL_NPLL>, <&cru SCLK_MAC>;
        assigned-clock-rates = <0>, <125000000>;
        phy-mode = "rgmii";
        pinctrl-names = "default";
        pinctrl-0 = <&rgmii_pins>;
        snps,reset-gpio = <&gpio3 RK_PB7 GPIO_ACTIVE_LOW>;
        snps,reset-active-low;
        snps,reset-delays-us = <0 10000 50000>;
        tx_delay = <0x2e>;
        rx_delay = <0x0f>;
        status = "okay";
};
```

1.3.3 Auto Scanning

If a set of delayline values cannot be adapted to all hardware, the reason may be poor hardware and a small window with poor redundancy; you can turn on the automatic function scanning and open `CONFIG_DWMAC_RK_AUTO_DELAYLINE` on menuconfig. It should be noted here that if the problem of the small window is not resolved, opening this macro will not completely solve the problem. Generally speaking, it is not necessary to open this macro.

```
    Device Drivers →
        Network device support →
            Ethernet driver support →
                [*]           Auto search rgmii delayline
```

This function will only make a detection at the first boot, and will store the delayline value to vendor storage after detection. Each subsequent boot will directly obtain the value from vendor storage and override the dts configuration. Only after the vendor storage is erased, the operation will be performed once after the next boot.

Log printing for the first boot:

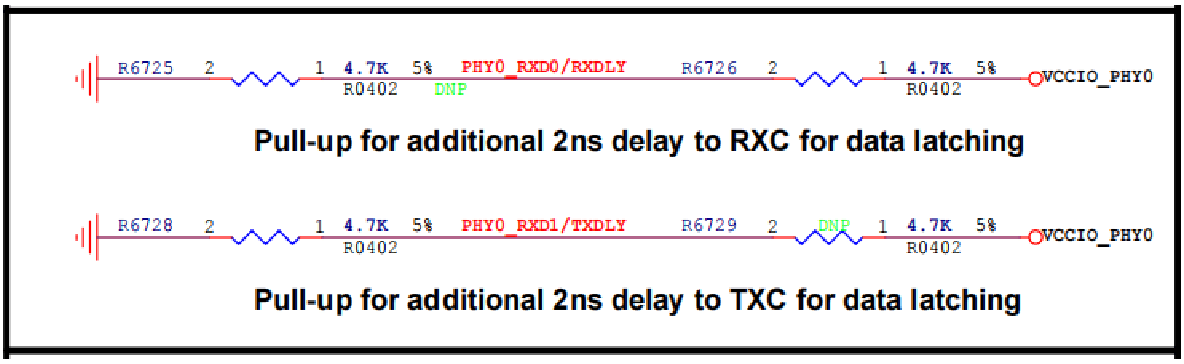
```
[   23.532138] Find suitable tx_delay = 0x2f, rx_delay = 0x10
```

Log printing for subsequent boot:

```
[   23.092358] damac rk read rgmii dl from vendor tx: 0x2f, rx: 0x10
```

1.4 rgmii-rxid 模式

When the hardware enables the RX delay of the PHY, such as RTL8211F:



Need to turn off the RX delay of the GMAC, and the dts configuration mode becomes "rgmii-rxid", for example:

```
&gmac0 {
    /* Use rgmii-rxid mode to disable rx delay inside Soc */
    phy-mode = "rgmii-rxid";
    clock_in_out = "output";

    ......
};
```

Scan through the above method to get:

```
# echo 1000 > phy_lb_scan
[  429.089639] Delayline scan speed set to 1000
[  429.560194] dwmac4: Master AXI performs any burst length
[  429.760434] RX(-01):0 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
[  429.859831] Find available tx_delay = 0x43, rx_delay = disable
[  430.089372] rk_gmac-dwmac fe1b0000.ethernet eth1: PHY [stmmac-0:01] driver [RTL8211F Gigabit Ethernet] (irq=POLL)
[  430.104205] dwmac4: Master AXI performs any burst length
[  430.104254] rk_gmac-dwmac fe1b0000.ethernet eth1: No Safety Features support found
[  430.104282] rk_gmac-dwmac fe1b0000.ethernet eth1: IEEE 1588-2008 Advanced Timestamp supported
[  430.104718] rk_gmac-dwmac fe1b0000.ethernet eth1: registered PTP clock
[  430.104758] rk_gmac-dwmac fe1b0000.ethernet eth1: configuring for phy/rgmii-rxid link mode
```

At this time, because the RX delay is fixed by the PHY hardware, usually 2ns, the RX delay will not be scanned. After the TX delay is scanned, the most suitable delay will be obtained. Fill in the "tx_delay" property in the dts gmac node, and comment to turn off rx_delay.

```
&gmac0 {
    /* Use rgmii-rxid mode to disable rx delay inside Soc */
    phy-mode = "rgmii-rxid";
    clock_in_out = "output";

    ......

    tx_delay = <0x43>;
    /* rx_delay = <0x42>; */

    phy-handle = <&rgmii_phy>;
    status = "okay";
};
```

2. Hareware Checking

2.1 Test RGMII Timing Specifics

According to the latest RGMII protocol, you need to meet the following timing requirements. Please test your board for compliance. If you know nothing about test or do not have an oscilloscope to test, please make a request on Redmine.

Symbol	Parameter	Min	Typical	Max	Units
TskewT	Data to Clock output Skew (at Transmitter) *note 1	-500	0	500	ps
TskewR	Data to Clock input Skew (at Receiver) *note 1	1	1.8	2.6	ns
TsetupT	Data to Clock output Setup (at Transmitter – integrated delay) *note 4	1.2	2.0		ns
TholdT	Clock to Data output Hold (at Transmitter – integrated delay) *note 4	1.2	2.0		ns
TsetupR	Data to Clock input setup Setup (at Receiver – integrated delay) *note 4	1.0	2.0		ns
TholdR	Data to Clock input setup Setup (at Receiver – integrated delay) *note 4	1.0	2.0		ns
Tcyc	Clock Cycle Duration *note 2	7.2	8	8.8	ns
Duty_G	Duty Cycle for Gigabit *note 3	45	50	55	%
Duty_T	Duty Cycle for 10/100T *note 3	40	50	60	%
Tr / Tf	Rise / Fall Time (20-80%)			.75	ns

note 1: For all versions of RGMII prior to 2.0; This implies that PC board design will require clocks to be routed such that an additional trace delay of greater than 1.5ns and less than 2.0ns will be added to the associated clock signal. For 10/100 the Max value is unspecified.
note 2: For 10Mbps and 100Mbps, Tcyc will scale to 400ns+-40ns and 40ns+-4ns respectively.
note 3: Duty cycle may be stretched/shrunk during speed changes or while transitioning to a received packet's clock domain as long as minimum duty cycle is not violated and stretching occurs for no more than three Tcyc of the lowest speed transitioned between.
Note 4: TsetupT / TholdT allows implementation of a delay on TXC or RXC inside the transmitter. Devices which implement internal delay shall be referred to as **RGMII-ID**. Devices may offer an option to operate with/without internal delay and still remain compliant with this spec.

For example, to confirm the signal quality of CLK at Gigabit, you should measure the waveforms of the signals at where close to the receiving end (do not measure at the sending end, the signal reflection at the sending end is serious, and the waveform cannot reflect the actual signal quality). Measuring the signal MAC_CLK, TX_CLK, RX_CLK, you should focus on duty cycle, amplitude, and rise and fall time, the bandwidth of the measuring oscilloscope and probe must be more than 5 times of 125M. For single-ended probes, the ground loop should be as short as possible. It is best to use a differential probe to control the duty cycle at 45 % ~ 55%. When the test environment is normal, the signal measured should be a square wave instead of a sine wave. Generally, the customer self-test is a positive sine wave with a 50% duty cycle, which is basically incorrect.

2.1.1 RX_CLK/MAC_CLK

MAC_CLK or RXCLK is provided by PHY. If the integrity of the received CLK measurement signal have defects, there is generally no register to regulate on the PHY side, it may only be adjusted by hardware. You can string high-frequency inductance at the transmitting end to improve the edge too slow （the bandwidth is only available if it meets the requirement so here cannot use ordinary inductance), and regulate the duty ratio and reduce the amplitude by modify voltage by resistance at the transmitting end.

2.1.2 TX_CLK

If there is a problem with TX_CLK, the edge is too slow, you can read the corresponding register through IO to check if the IO drive strength has been adjusted to the maximum, you can connect it with an oscilloscope; at the same time, directly regulate the drive strength through the IO command to observe the waveform change. If the improvement is not obvious by regulating driver, you can try for the string high-frequency inductance, or change and increase the series 22ohm resistor; if the duty cycle is not within the specification and TX_CLK bypass from MAC_CLK, you can adjust the TX_CLK duty cycle by dividing the MAC_CLK amplitude, the divided value is 100 ohm, the ground resistance value varies with the layout, different board have different value, you can regulate up from 100, until the oscilloscope observes that the duty cycle meets the requirements. If the above methods are not significant, and the IO currently used is 3.3V, in the case that both the PHY and RK platform support 1.8V

IO, you can change the IO power to 1.8V and then observe the signal integrity because 1.8V IO signal indicator stronger than 3.3V, 1.8V IO is recommended.

3. FAQ

3.1 Scanning Window size

We hope that the larger the window, the better, indicating that the hardware signal is good and the redundancy is large. If the window cannot be scanned or the window scanned is too small, there are generally hardware problems, please refer to the hardware section.

3.2 PHY Selection

There are no special requirements for the selection of PHY, as long as it conforms to RGMII. However, there are two points to note:

- If your project plans to use the RTL8211E Gigabit PHY, it is recommended that you change to RTL8211F or other PHY, for reasons explained in section 2.1 above.
- If the PHY you are using does not have a loopback function, please refer to the following method to obtain the delayline:

  PHY can be debugged based on the oscilloscope measurement signal. Use an oscilloscope with a bandwidth greater than 125M and 5 times the bandwidth to measure the phase difference between TXC and TXD near the PHY end. Regulate the phase within the range of 1.5ns ~ 2ns by IO command (the specification is 1 ~ 2.6 ns should be left in a certain amount), TX will be fine; and due to RX delayed inside the SoC, as loopback is not used, it can only be judged with the help of throughput. When the tx_delay register is determined, firstly set rx_delay to 0x10. After the change, use iperf to run the down link throughput. The result may be unsatisfactory at the beginning, continue to use the IO command to change the register rxdelay and increase it in steps of 5 (0x10 ~ 0x7f interval). After the IO writes the register, when it is measured that the throughput is greater than 900M, then narrow down by 2 steps to find the register interval that can go up to 900M, and then take the intermediate value and set it to dts.

TXC&TXD phase waveform figure：