DDR Develop Guide

ID: RK-KF-YF-40

Release Version: V1.4.0

Release Date: 2022-05-06

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

This document introduces the double data rate(DDR) SDRAM develop work, which is suitable to all Rockchip chips.

Overview

Product ID

Chipset Name	Kernel Version
All chipset	All kernel version

Intended Audience

This document (this guide) is mainly intended for:

Technical support engineers

Software development engineers

Revision History

Date	Revision No.	Author	History
2017.12.21	V1.0.0	CanYang He	
2018.3.30	V1.1.0	CanYang He	Added the related description of Kernel 4.4 DDR frequency
2019.1.29	V1.2.0	Zhihuan He	Added the statement on adjusting the de-skew in loader
2021.1.21	V1.3.0	YunPing Tang	Added the statement for RV1126/RV1109/RK356x
2022.5.6	V1.4.0	Zhihuan He	Added the statement for RK3326S/PX30S

Contents

What the Meaning of DDR log

The DDR log includes the log in the loader and the log in the kernel. The log in the loader is parsed as follows :

```
DDR Version 1.05 20170712//Version information of the DDR initialization code
used to check the version. From this line, you have entered the DDR
initialization code.
In
SRX //If it prints SRX, means hot restart; without SRX, it means that it is cold
boot. While some chipset does not have this feature, there will not show SRX.
Channel a: DDR3 400MHz //The following log are the details of the DDR capacity.
For more explanation,please see the chapter"How to Check the Capacity of DDR".
Bus Width=32 Col=10 Bank=8 Row=15 CS=1 Die Bus-Width=16 Size=1024MB
Channel b: DDR3 400MHz
Bus Width=32 Col=10 Bank=8 Row=15 CS=1 Die Bus-Width=16 Size=1024MB
Memory OK //This is the result of DDR self-test, the first "Memroy OK" is the
self-test result of Channel a.
Memory OK //It is the self-test result of Channel b.If Channel a or b shows an
error, turning out that something wrong with the welding; no error, indicating
that the current self-test is good.But whether the entire DDR can work stably or
not,also depends on the subsequent stages of operation results.
OUT //After this line, the DDR initialization code is exited.
```

Below is the DDR log of kernel 3.0 and kernel 3.10:

```
[    0.528564] DDR DEBUG: version 1.00 20150126 //Version information
[    0.528690] DDR DEBUG: Channel a:    //The details of the DDR capacity
[    0.528701] DDR DEBUG: DDR3 Device
[    0.528716] DDR DEBUG: Bus Width=32 Col=10 Bank=8 Row=15 CS=1 Total
Capability=1024MB
[    0.528727] DDR DEBUG: Channel b:
[    0.528736] DDR DEBUG: DDR3 Device
[    0.528750] DDR DEBUG: Bus Width=32 Col=10 Bank=8 Row=15 CS=1 Total
Capability=1024MB
//The following information about DDR specialize for DDR engineer debug, please
ignore it.
//After "DDR DEBUG" print end, which means DDR initialization finishes in
kernel.
```

The kernel 3.10 will also have the following log, which is the output information of the DDR frequency scaling module.

```
[    1.473637] ddrfreq: verion 1.2 20140526 //DDR frequency scaling module
version
[    1.473653] ddrfreq: normal 396MHz video_1080p 240MHz video_4k 396MHz
dualview 396MHz idle 0MHz suspend 200MHz reboot 396MHz //The frequencies which
read from dts table are corresponding to the different scenarios.
[    1.473661] ddrfreq: auto-freq=1 //This line reflects load scaling functon is
enable or not,"1" means on,"0" means off.
[    1.473667] ddrfreq: auto-freq-table[0] 240MHz   //the  table of the load
scaling
[    1.473673] ddrfreq: auto-freq-table[1] 324MHz
[    1.473678] ddrfreq: auto-freq-table[2] 396MHz
[    1.473683] ddrfreq: auto-freq-table[3] 528MHz
//If crash or block in this print porcedure,it is most likely DDR frequency
scaling bug.
```

How to Integrate RK DDR Bin into A Completed and Usable Loader

1. Put the DDR bin in the corresponding directory of the `rk\rkbin\bin\` of the U-Boot project.
2. Delete the original DDR bin file.
3. Rename the new DDR bin to the name which have been deleted.
4. Compile U-Boot (see "Rockchip-Developer-Guide-UBoot-nextdev.pdf"), it will generate the corresponding loader file.
5. Confirm that the loader already updated correctly according to the log of loader

Summarize all platforms DDR bin corresponding directory as below:

Chip Type	Path	Note
RK1808	rk\rkbin\bin\rk1x\rk1808_ddr_XXXMHz_vX.XX.bin	
RK3036	rk\rkbin\bin\rk30\rk3036_ddr3_XXXMHz_vX.XX.bin	1
RK3126、 RK3126B、 RK3126C	rk\rkbin\bin\rk31\rk3126_ddr3_300MHz_vX.XX.bin	
RK3128	rk\rkbin\bin\rk31\rk3128_ddr_300MHz_vX.XX.bin	
RK3288	rk\rkbin\bin\rk32\rk3288_ddr_400MHz_vX.XX.bin	
RK322x	rk\rkbin\bin\rk32\rk322x_ddr_300MHz_vX.XX.bin	
RK3308	rk\rkbin\bin\rk33\rk3308_ddr_XXXMHz_uartX_mX_vX.XX.bin	
PX30	rk\rkbin\bin\rk33\px30_ddr_333MHz_vX.XX.bin	
RK3326	rk\rkbin\bin\rk33\rk3326_ddr_333MHz_vX.XX.bin	
RK3368	rk\rkbin\bin\rk33\rk3368_ddr_600MHz_vX.XX.bin	
RK322xh	rk\rkbin\bin\rk33\rk322xh_ddr_333MHz_vX.XX.bin	
RK3328	rk\rkbin\bin\rk33\rk3328_ddr_333MHz_vX.XX.bin	
RK3399	rk\rkbin\bin\rk33\rk3399_ddr_XXXMHz_vX.XX.bin	2

Note 1: To use which frequency is specified in `rk\rkbin\RKBOOT\RK3036_ECHOMINIALL.ini` or `RK3036MINIALL.ini`. And `RK3036_ECHOMINIALL.ini` is special for ECHO products, the other RK3036 products use RK3036MINIALL.ini. As for how to check ECHO machine, please consult Rockchip system product department.

Note 2: To use which frequency is specified in `rk\rkbin\RKBOOT\RK3399MINIALL.ini` file.

Note 3: The chipsets not involved in this table, may not support generating loaders from U-Boot.

How to Change DDR Frequency in U-Boot

Currently RK322x supports this feature only. The method is to modify `arch/arm/boot/dts/rk322x.dtsi` in kernel-3.10 code.

```
dram: dram {
    compatible = "rockchip,rk322x-dram";
    status = "okay";
    dram_freq = <786000000>;
    rockchip,dram_timing = <&dram_timing>;
};
```

You just need to modify "dram_freq" in the above code block and unit here is Hz. The frequency can be selected freely.

U-Boot will parse this DTS automatically, then read and scale it to the corresponding frequency.

How to Enable/Disable the DDR Frequency Scaling Function in the Kernel

Firstly, confirm that the chip do support DDR frequency scaling in the kernel. After that, you can enable or disable frequency scaling feature as follow method:

- For kernel 4.4, you need to find the final **dmc** node in dts. Change the status to "disabled" to disable the DDR scaling function in the kernel. Conversely, changing to "okay" will enable DDR frequency scaling.

  Note: It is better keep **dfi** node status consistent with **dmc** node because **dmc** node restricted by **dfi** node in the lagacy code, **dfi** node "disabled " would make the **dmc** node invalid.

  For example, RK3399 EVB, the final **dmc** node is in `arch/arm64/boot/dts/rockchip/rk3399-evb.dtsi`.

```
&dfi {
    status = "okay";
};

&dmc {
    status = "okay"; /* enable kernel DDR scaling function */
    ........
};
```

```
&dfi {
    status = "disabled";
};

&dmc {
    status = "disabled";    /* disable kernel DDR scaling function */
    ........
};
```

- For kernel 3.10, you need to find the final **clk_ddr_dvfs_table** node in dts. Modify the status to "disabled" to disable the DDR scaling function in the kernel. Conversely, modify to "okay "will enable the DDR scaling function.

  For example, the final `clk_ddr_dvfs_table` of the RK3288 SDK board is in `arch/arm/boot/dts/rk3288-tb_8846.dts.`

```
&clk_ddr_dvfs_table {
    ........
    status="okay";  /* enable kernel DDR scaling function */
};
```

```
&clk_ddr_dvfs_table {
    ........
    status="disabled";  /* disable kernel DDR scaling function */
};
```

- For kernel 3.0, you need to modify `dvfs_ddr_table` in the board-level `borad-**.c` file, leaving only one `DDR_FREQ_NORMAL` frequency in the table, so that DDR cannot change frequency.

  For example, the board file of the RK3066 SDK board is in `arch/arm/mach-rk30/board-rk30-sdk.c` as below:

```
/* This table enable DDR scaling function */
static struct cpufreq_frequency_table dvfs_ddr_table[] = {
    {.frequency = 200 * 1000 + DDR_FREQ_SUSPEND, .index = 1050 * 1000},
    {.frequency = 300 * 1000 + DDR_FREQ_VIDEO, .index = 1050 * 1000},
    {.frequency = 400 * 1000 + DDR_FREQ_NORMAL, .index = 1125 * 1000},
    {.frequency = CPUFREQ_TABLE_END},
};
```

```
/* This table disable DDR scaling function */
static struct cpufreq_frequency_table dvfs_ddr_table[] = {
    //{.frequency = 200 * 1000 + DDR_FREQ_SUSPEND, .index = 1050 * 1000},
    //{.frequency = 300 * 1000 + DDR_FREQ_VIDEO, .index = 1050 * 1000},
    {.frequency = 400 * 1000 + DDR_FREQ_NORMAL, .index = 1125 * 1000},
    {.frequency = CPUFREQ_TABLE_END},
};
```

How to Prohibit DDR Scaling include in initialization state

The previous topic just talk about how to enable or disable DDR scaling function ,keeping you machine running without scaling.But there is a exception in initialization,DDR will scale frequency once in `ddr_init` when you power on, to update DDR timing for higher performance.So if you need disable DDR scaling function include in `ddr_init`, you need modify code referred to Chapter "How to Enable/Disable the DDR Frequency Scaling Function in the Kernel" **and** the code below:

- For kernel 4.4

  Only following the Chapter "How to Enable/Disable the DDR Frequency Scaling Function in the Kernel",DDR frequency scaling will stop working, included in `ddr_init`.

- For kernel 3.10

  Chip Type:**RK322X**

  Code Location: NO code in kernel

  Method: Modify dram node to "disabled" only

```
dram: dram {
    compatible = "rockchip,rk322x-dram";
    status = "disabled";    /* Please,modify here! */
    dram_freq = <786000000>;
    rockchip,dram_timing = <&dram_timing>;
};
```

Chip type : **RK3188**

Code Location: ddr_init() function in the file `arch/arm/mach-rockchip/ddr_rk30.c`

Chip type : **RK3288**

Code Location: ddr_init() function in the file `arch/arm/mach-rockchip/ddr_rk32.c`

Chip type : **RK3126B、RK3126C which firmware without `trust.img`**

Code Location: ddr_init() function in the file `arch/arm/mach-rockchip/ddr_rk3126b.c`

Chip type : **RK3126/RK3128**

Code Location: ddr_init() function in the file `./arch/arm/mach-rockchip/ddr_rk3126.c`

Method: comment out the following lines in `ddr_init()` function code :

```
if(freq != 0)
    value = clk_set_rate(clk, 1000*1000*freq);
else
    value = clk_set_rate(clk, clk_get_rate(clk));
```

Chip type : **RV1108**

Code Location: ddr_init() function in the file `arch/arm/mach-rockchip/ddr_rv1108.c`

Method: comment out the following lines in `ddr_init()` function code :

```
if (freq == 0)
_ddr_change_freq(ddr_freq_current);
else
_ddr_change_freq(freq);
```

The other chip, included RK3126B and RK3126C which firmware with trust.img, only need to do following the Chapter "How to Enable/Disable the DDR Frequency Scaling Function in the Kernel", DDR frequency scaling will stop working, included in `ddr_init`.

- For kernel 3.0

Chip Type	Code Path
RK3066	arch/arm/mach-rk30/ddr.c, ddr_init() function
RK3026、RK3028A	arch/arm/mach-rk2928/ddr.c, ddr_init() function

Method: comment out the following lines in `ddr_init()` function code :

```
if(freq != 0)
    value=ddr_change_freq(freq);
else
    value=ddr_change_freq(clk_get_rate(clk_get(NULL, "ddr"))/1000000);
```

How to Check the DDR Capacity

If you look for a DDR capacity roughly, using the command blow.This data looks a little smaller than real, please estimate it to an integer value.

```
root@rk3399:/ # cat /proc/meminfo
MemTotal:        3969804 kB
```

If you need for more detail about DDR capacity, follow this:

DDR capacity printing in 2 places,which is in DDR initialization stage in loader and kernel. There is no DDR capacity information to print in kernel 4.4 while some chip have these in kernel 3.10(see the table below). The DDR capacity details in the loader are available on all chips. The DDR capacity printing in the loader must be captured by the serial port, if using ADB, you will miss this part.

Chip Type	loader	kernel 3.0/3.10
RK3026	√	√
RK3028A	√	√
RK3036	√	×
RK3066	√	√
RK3126B、RK3126C with trust.img	√	×
RK3126B、RK3126C without trust.img	√	√
RK3126	√	√
RK3128	√	√
RK3188	√	√
RK3288	√	√
RK322x	√	×
RK322xh	√	×
RK3328	√	×
RK3368	√	×
RK3399	√	×
RV1108	√	×

√ means have capacity printing

× means no capacity printing

The DDR detail contains:DDR type/DDR frequency/Channel (channel a/ channel b)/bus width(BW)/row/column(col)/bank(BK)/CS/die bus width(die BW)/size (total capability)

The whole capacity equals to size/ total capacity when SOC chip only has 1 DDR channel or the sum of two channel's size/total capacity.

The detail of DDR capacity in the loader as below:

```
DDR Version 1.05 20170712
In
Channel a: DDR3 400MHz
Bus Width=32 Col=10 Bank=8 Row=15 CS=1 Die Bus-Width=16 Size=1024MB
Channel b: DDR3 400MHz
Bus Width=32 Col=10 Bank=8 Row=15 CS=1 Die Bus-Width=16 Size=1024MB
Memory OK
Memory OK
OUT
```

The detail of DDR capacity in the kernel as below:

```
[    0.528564] DDR DEBUG: version 1.00 20150126
[    0.528690] DDR DEBUG: Channel a:
[    0.528701] DDR DEBUG: DDR3 Device
[    0.528716] DDR DEBUG: Bus Width=32 Col=10 Bank=8 Row=15 CS=1 Total
Capability=1024MB
[    0.528727] DDR DEBUG: Channel b:
[    0.528736] DDR DEBUG: DDR3 Device
[    0.528750] DDR DEBUG: Bus Width=32 Col=10 Bank=8 Row=15 CS=1 Total
Capability=1024MB
[    0.528762] DDR DEBUG: addr=0xd40000
```

How to Modify DDR Frequency

There are 2 strategies in the kernel:scenario frequency scaling and loading frequency scaling.The operation between kernel 4.4 and kernel 3.10 has some difference.

kernel 4.4:

Scenario frequency scaling means: entered the specified scenario, DDR frequency will change to the corresponding frequency defined by `SYS_STATUS_XXX` if the load frequency scaling function disabled.In the contrary, load frequency scaling function is enable, it will increase or reduce frequency based on the actual DDR status and the defined value of `upthreshold/downdifferential`, but frequency will not be lower than the value from `SYS_STATUS_XXX`.

Load frequency scaling means: The frequency depends on the load status in all scenario, but higher than the defined value from `SYS_STATUS_XXX`.Only the special `SYS_STATUS_NORMAL` is replaced by load frequency value, and the lowest frequency was controlled by `auto-min-freq` instead of `SYS_STATUS_NORMAL`.

kernel 3.10:

Scenario frequency scaling means: Entered the specific scenario, DDR frequency change to the value of `SYS_STATUS_XXX` and no more change though the load frequency scaling function is enabled.

Load frequency scaling means: it is used to replace scenario `SYS_STATUS_NORMAL`, DDR frequency depends on the load status only in `SYS_STATUS_NORMAL`.

To modify the DDR frequency, it still has to be handled by kernel branch separately.

- For kernel 4.4, it requires get the **dmc** node in dts. For example, **dmc** node in RK3300 EVB is in `arch/arm64/boot/dts/rockchip/rk3399-evb.dtsi` and `arch/arm64/boot/dts/rockchip/rk3399.dtsi`

```
&dmc {
    status = "okay";
    center-supply = <&vdd_center>;
    upthreshold = <40>;
    downdifferential = <20>;
    system-status-freq = <
        /*system status         freq(KHz)*/
        SYS_STATUS_NORMAL       800000
        SYS_STATUS_REBOOT       528000
        SYS_STATUS_SUSPEND      200000
        SYS_STATUS_VIDEO_1080P  200000
        SYS_STATUS_VIDEO_4K     600000
        SYS_STATUS_VIDEO_4K_10B 800000
        SYS_STATUS_PERFORMANCE  800000
        SYS_STATUS_BOOST        400000
        SYS_STATUS_DUALVIEW     600000
        SYS_STATUS_ISP          600000
    >;
    /* Each line is used as a group of data, "min_bw "and "max_bw" represent the
bandwidth requirement corresponded by vop.When the requirement value fallling
between the range of "min_bw" and "max_bw", the DDR frequency needs to increase
the frequency specified by "freq", and is valid at "auto-freq-en=1" */
    vop-bw-dmc-freq = <
    /* min_bw(MB/s) max_bw(MB/s) freq(KHz) */
        0       577       200000
        578     1701      300000
        1702    99999     400000
    >;
    auto-min-freq = <200000>;
};
```

```
dmc: dmc {
    compatible = "rockchip,rk3399-dmc";
    devfreq-events = <&dfi>;
    interrupts = <GIC_SPI 1 IRQ_TYPE_LEVEL_HIGH 0>;
    clocks = <&cru SCLK_DDRCLK>;
    clock-names = "dmc_clk";
    ddr_timing = <&ddr_timing>;
    /* DDR utilization exceeds 40%, starts to increase frequency when "auto-
freq-en=1 " */
    upthreshold = <40>;
    /* DDR utilization less than 20%, start to reduce frequency when "auto-freq-
en=1 " */
    downdifferential = <20>;
    system-status-freq = <
    /*system status         freq(KHz)*/
    /* It is valid when "auto-freq-en=0". It indicates that this scene is in
common use except for the following scenes */
    SYS_STATUS_NORMAL       800000
    /* It means the DDR frequency before reboot. When auto-freq-en=1, this
frequency will be used as the min value and increased according to the load
status */
```

```
    SYS_STATUS_REBOOT       528000
    /* It means the DDR frequency at early suspend. When auto-freq-en=1, this
frequency will be used as the min value and increased according to the load
status */
    SYS_STATUS_SUSPEND      200000
    /* It means the DDR frequency at playing 1080P video.When auto-freq-en=1,
this frequency will be used as the min value and increased according to the load
status */
    SYS_STATUS_VIDEO_1080P  300000
    /* It means the DDR frequency at playing 4K video When auto-freq-en=1, this
frequency will be used as the min value and increased according to the load
status */
    SYS_STATUS_VIDEO_4K     600000
    /* It means the DDR frequency at playing 4K 10bit video. When auto-freq-
en=1, this frequency will be used as the min value and increased according to
the load status */
    SYS_STATUS_VIDEO_4K_10B 800000
    /* It means the DDR frequency at performance mode.When auto-freq-en=1, this
frequency will be used as the min value and increased according to the load
status */
    SYS_STATUS_PERFORMANCE  800000
    /* It means the DDR frequency at touching,getting higher frequency from low
in order to improve touching respond.When auto-freq-en=1, this frequency will be
used as the min value and increased according to the load status */
    SYS_STATUS_BOOST        400000
    /* It means the DDR frequency at dual display mode.When auto-freq-en=1, this
frequency will be used as the min value and increased according to the load
status */
    SYS_STATUS_DUALVIEW     600000
    /* It means the DDR frequency at ISP mode.When auto-freq-en=1, this
frequency will be used as the min value and increased according to the load
status */
    SYS_STATUS_ISP          600000
    >;
    /* When auto-freq-en=1, this frequency will be used as the min value of
SYS_STATUS_NORMAL scenario */
    auto-min-freq = <400000>;
    /* The value equals to 1,which indicates this function is on, to 0,which
means off.If it is on, "SYS_STATUS_NORMAL" will be taken by the load frequency
completely and the lowest frequency is" auto-min-freq" instead of
"SYS_STATUS_NORMAL".That means,it takes the frequency defined by this scene as
the lowest frequency and the system will increase or reduece DDR frequency
through "upthreshold/downdifferential" according to DDR utilization */
    auto-freq-en = <1>;
    status = "disabled";
};
```

==Note==: Kernel 4.4 frequency voltage is different from kernel 3.10, it runs in this frequency only when frequency equals to `opp-hz` listed by `dmc_opp_table`.If the frequency less than `opp-hz`, compatible to it upwardly,otherwise, it exceeds `opp-hz` the upper limited,it will restricted by `opp-hz`.So, if you do not want to be controlled, you should concern `dmc_opp_table`.

```
dmc_opp_table: opp-table3 {
    opp-200000000 {
        /* When the DDR frequency equals to 200MHz,this voltage is
effective;less than 200MHz,running at 200MHz */
        opp-hz = /bits/ 64 <200000000>;
        opp-microvolt = <825000>;   //vdd_center voltage
    };
    ......
    opp-800000000 {
        opp-hz = /bits/ 64 <800000000>;
        opp-microvolt = <900000>;
    };
};
```

After understanding the meaning of each configuration, modify the corresponding frequency definition according to the scene you need to modify. If `auto-freq-en=1`, it is not good to control the frequency. If reducing frequency is to locate problem, you can set `auto-freq-en` value to 0, then modify the frequency value defined by each scene to achieve your purpose.

- To kernel3.10, it requires to find the node `clk_ddr_dvfs_table` in dts. For example, RK3288 SDK's last node `clk_ddr_dvfs_table` is in `arch/arm/boot/dts/rk3288-tb_8846.dts`.

```
&clk_ddr_dvfs_table {
    /* The logic voltage corresponding to the DDR frequency, if the frequency in
"freq-table" or "bd-freq-table" is larger than the maximum frequency here, the
corresponding voltage cannot be found and can not switched to the corresponding
frequency. At this time, you need to add frequency voltage table here */
    operating-points = <
        /* KHz    uV */
        200000 1050000
        300000 1050000
        400000 1100000
        533000 1150000
        >;

    freq-table = <
        /*status        freq(KHz)*/
        /* It is valid only when "auto-freq-en=0".And it indicates that this
scene is common use scene except for the following scenes */
        SYS_STATUS_NORMAL    400000
        /* DDR frequency at the early suspend */
        SYS_STATUS_SUSPEND  200000
        /* DDR frequency at playing 1080P video */
        SYS_STATUS_VIDEO_1080P  240000
        /* DDR frequency at playing 4K video */
        SYS_STATUS_VIDEO_4K     400000
        /* DDR frequency at playing 60FPS video */
        SYS_STATUS_VIDEO_4K_60FPS     400000
        /* DDR frequency at performance mode */
        SYS_STATUS_PERFORMANCE  528000
        /* DDR frequency at dual display */
        SYS_STATUS_DUALVIEW 400000
        /* DDR frequency at touching,getting higher frequency from low in order
to improve touching respond */
        SYS_STATUS_BOOST    324000
        /* DDR frequency at ISP  */
```

```
        SYS_STATUS_ISP      400000
        >;
    bd-freq-table = <
        /* bandwidth   freq */
        5000           800000
        3500           456000
        2600           396000
        2000           324000
    >;
    /* After the load frequency scaling turned on,where the "SYS_STATUS_NORMAL"
scenario, it will switch between several frequencies listed by this table
according to the DDR bandwidth utilization */
    auto-freq-table = <
        240000
        324000
        396000
        528000
        >;
    /* The value equals to "1", indicating that the load frequency conversion
function is enabled; equals to 0, means disabled. After the load frequency
conversion function turning on, the "SYS_STATUS_NORMAL" scene frequency scaling
will be completely replaced by the load scaling frequency */
    auto-freq=<1>;
    /*
     * 0: use standard flow
     * 1: vop dclk never divided
     * 2: vop dclk always divided
     */
    vop-dclk-mode = <0>;
    status="okay";
};
```

After understanding the meaning of each configuration, modify the corresponding frequency definition according to the scene you need to modify. If `auto-freq-en=1`, it is not good to control the frequency. If reducing frequency is to locate problem, you can set `auto-freq-en` value to 0, then modify the frequency value defined by each scene to achieve your purpose.

==Note: you must make sure that the voltage can work at this frequency==.As for how to modify voltage, see the chapter "How to modify the voltage corresponding to a certain DDR frequency ".

- To kernel3.10, it requires to find the `dvfs_ddr_table` in board document `borad-**.c`. For example, RK3066 SDK's `dvfs_ddr_table` is in `arch/arm/mach-rk30/board-rk30-sdk.c`.

```
static struct cpufreq_frequency_table dvfs_ddr_table[] = {
    /* DDR frequency at the early suspend */
    {.frequency = 200 * 1000 + DDR_FREQ_SUSPEND, .index = 1050 * 1000},
    /* DDR frequency at playing video */
    {.frequency = 300 * 1000 + DDR_FREQ_VIDEO, .index = 1050 * 1000},
    /* it indicates that this scene is common use scene except for above two
scenes */
    {.frequency = 400 * 1000 + DDR_FREQ_NORMAL, .index = 1125 * 1000},
    {.frequency = CPUFREQ_TABLE_END},
};
```

Kernel 3.0 has only 3 scenes. The DDR frequency to be modified is in `"200 * 1000"` of `.frequency` and the frequency unit here is KHz. The `"+ DDR_FREQ_SUSPEND"` string can be ignored.

==Note: you must make sure that the voltage can work at this frequency==.As for how to modify voltage, see the chapter "How to modify the voltage corresponding to a certain DDR frequency ".

How to Modify the Voltage Corresponding to A Certain DDR Frequency

If you want to locate bug through changing the voltage by command, use the following method:

kernel 4.4: You need to compile the kernel, select "pm_tests" option (make ARCH=arm64 menuconfig ->Device Drivers -> SOC (System On Chip) specific Drivers -> Rockchip pm_test support)

kernel 3.10：You need to compile the kernel, open "pm_tests" option (make menuconfig ->System Type -> /sys/pm_tests/ support)。

The command to modify the DDR voltage is：

RK3399： `echo set vdd_center  900000 > /sys/pm_tests/clk_volt`

Other Chip： `echo set vdd_logic  1200000 > /sys/pm_tests/clk_volt`

If there is no "pm_tests" or the command cannot meet the requirements, you need to change the kernel firmware, as follows:

- For kernel 4.4, you need to find the node `dmc_opp_table` in dts. For example,RK3399 EVB's node is in `arch/arm64/boot/dts/rockchip/rk3399-opp.dtsi`,RK3368's node is in `arch/arm64/boot/dts/rockchip/rk3368.dtsi`

  RK3399:

```
/* it runs in this frequency only when frequency equals to "opp-hz"listed by
"dmc_opp_table".If the frequency less than "opp-hz", the frequency will getting
higher,otherwise, it exceeds "opp-hz" the upper limited,it will restricted by
"opp-hz".It is different from kernel 3.10 */
dmc_opp_table: opp-table3 {
    compatible = "operating-points-v2";

    opp-200000000 {
        /* When the DDR frequency equals to 200MHz,this voltage is
effective;less than 200MHz,running at 200MHz */
        opp-hz = /bits/ 64 <200000000>;
        opp-microvolt = <825000>;   //vdd_center voltage
    };
    opp-300000000 {
        opp-hz = /bits/ 64 <300000000>;
        opp-microvolt = <850000>;
    };
    opp-400000000 {
        opp-hz = /bits/ 64 <400000000>;
        opp-microvolt = <850000>;
    };
    opp-528000000 {
        opp-hz = /bits/ 64 <528000000>;
        opp-microvolt = <900000>;
```

```
    };
    opp-600000000 {
        opp-hz = /bits/ 64 <600000000>;
        opp-microvolt = <900000>;
    };
    opp-800000000 {
        opp-hz = /bits/ 64 <800000000>;
        opp-microvolt = <900000>;
    };
};
```

Take RK3368 as an example:

```
/* it runs in this frequency only when frequency equals to "opp-hz"listed by
"dmc_opp_table".If the frequency less than "opp-hz", the frequency will getting
higher,otherwise, it exceeds "opp-hz" the upper limited,it will restricted by
"opp-hz".It is different from kernel 3.10 */
dmc_opp_table: opp_table2 {
    compatible = "operating-points-v2";

    opp-192000000 {
        /* When the DDR frequency equals to 200MHz,this voltage is
effective;less than 200MHz,running at 200MHz */
        opp-hz = /bits/ 64 <192000000>;
        opp-microvolt = <1100000>; //vdd_logic voltage
    };
    opp-300000000 {
        opp-hz = /bits/ 64 <300000000>;
        opp-microvolt = <1100000>;
    };
    opp-396000000 {
        opp-hz = /bits/ 64 <396000000>;
        opp-microvolt = <1100000>;
    };
    opp-528000000 {
        opp-hz = /bits/ 64 <528000000>;
        opp-microvolt = <1100000>;
    };
    opp-600000000 {
        opp-hz = /bits/ 64 <600000000>;
        opp-microvolt = <1100000>;
    };
};
```

The voltage in accordance with the frequency can be modified. Since the frequency-voltage table using voltage less than or equal to the specified frequency, the added frequency that exceeds the limited frequency of this table cannot match the appropriated voltage, which will cause DDR fail to switch to the new frequency. At this time, it is necessary to add a frequency-voltage item corresponding to the frequency.

- For kernel 3.10, you need to find the node `clk_ddr_dvfs_table` in dts , for example, RK3288 SDK the last `clk_ddr_dvfs_table` is in `arch/arm/boot/dts/rk3288-tb_8846.dts`.

```
&clk_ddr_dvfs_table {
    /* This is Frequency-voltage table */
    operating-points = <
```

```
        /* KHz    uV */
        /* it is show when DDR frequency less than or equals to 200MHz,logic
voltage uses 1050mV.Other lines mean the same here */
        200000 1050000
        300000 1050000
        400000 1100000
        533000 1150000
        >;

    .......
    status="okay";
};
```

The voltage in accordance with the frequency can be modified. Since the frequency-voltage table using voltage less than or equal to the specified frequency, the added frequency that exceeds the limited frequency of this table cannot match the appropriated voltage, which will cause DDR fail to switch to the new frequency. At this time, it is necessary to add a frequency-voltage item corresponding to the frequency.

- For kernel 3.0, you need to modify `dvfs_ddr_table` in the file `borad-**.c` ,for example, RK3066 SDK's is in `arch/arm/mach-rk30/board-rk30-sdk.c`.

```
static struct cpufreq_frequency_table dvfs_ddr_table[] = {
    {.frequency = 200 * 1000 + DDR_FREQ_SUSPEND, .index = 1050 * 1000},
    {.frequency = 300 * 1000 + DDR_FREQ_VIDEO, .index = 1050 * 1000},
    {.frequency = 400 * 1000 + DDR_FREQ_NORMAL, .index = 1125 * 1000},
    {.frequency = CPUFREQ_TABLE_END},
};
```

The `".index"` in the `dvfs_ddr_table` is the corresponding voltage,unit here is uV.

How to Disable the Load DDR Frequency Scaling with Leaving Only the Scene Frequency Scaling

- For kernel 4.4 ,you need to find `auto-freq-en` of the **dmc** node in dts.For example, RK3399 EVB's `auto-freq-en` is in `arch/arm64/boot/dts/rockchip/rk3399.dtsi`.

```
dmc: dmc {
    compatible = "rockchip,rk3399-dmc";
    ........
    auto-min-freq = <400000>;
    /* Set this value to 0 to close the load DDR Frequency scaling  with leaving
only the scene frequency scaling */
    auto-freq-en = <0>;
    .......
};
```

- For kernel 3.10 ,you need to find the node `clk_ddr_dvfs_table` in dts, For example, RK3288 EVB's `clk_ddr_dvfs_table` is in `arch/arm/boot/dts/rk3288-tb_8846.dts`

```
&clk_ddr_dvfs_table {
    .......
    /* Set this value to 0 to close the load DDR Frequency scaling  with leaving
only the scene frequency scaling */
    auto-freq=<0>;
    ......
    status="okay";
};
```

- Kernel 3.0 itself does not support the load frequency scaling, let alone closing it.

How to Fix DDR Frequency

If you want to locate bug through fixing DDR frequency by command, use the following method:

kernel 4.4:

Get the available DDR frequency:
`cat /sys/class/devfreq/dmc/available_frequencies`

Set frequency:

`echo userspace > /sys/class/devfreq/dmc/governor`
`echo 300000000 > /sys/class/devfreq/dmc/min_freq` //This line purposes to prevent the frequency to be set lower than "min_freq", cause operation failed.
`echo 300000000 > /sys/class/devfreq/dmc/userspace/set_freq`

kernel 3.10:

You need to compile the kernel, open "pm_tests" option (make menuconfig ->System Type -> /sys/pm_tests/ support) , Fixing DDR frequency command is

`echo set clk_ddr 300000000 > /sys/pm_tests/clk_rate`

The frequency unit here is Hz and the command parameter can be changed according to the requirement.

If the method above is not feasible, you can only modify the code or dts.

- For kernel 4.4, if the method above does not work, it is generally because the target frequency, not in `cat /sys/class/devfreq/dmc/available_frequencies`.

  The way to solve this problem is to find the board-level dts file and add your target frequency in `dmc_opp_table`. For example, the RK3399 EVB board is in `arch/arm64/boot/dts/rockchip/rk3399-opp.dtsi`.Here assuming you want to add 666MHz:

  ```
  dmc_opp_table: opp-table3 {
  compatible = "operating-points-v2";

  opp-200000000 {
      opp-hz = /bits/ 64 <200000000>;
      opp-microvolt = <825000>;
  };
  ......
  opp-666000000 {
      /* When DDR frequency equals to 666MHz,use this voltage */
      opp-hz = /bits/ 64 <666000000>;
  ```

```
    opp-microvolt = <900000>;    //vdd_center voltage
};
opp-800000000 {
    opp-hz = /bits/ 64 <800000000>;
    opp-microvolt = <900000>;
};
};
```

After that, you can just use the previous command to fix the frequency.

If you do not want to fix frequency through inputing command at power-on, but starts from at a fixed frequency, modify the dts as beblow:

Supposed your target frequency is 666MHz. For example, the **dmc** node of RK3399 EVB board is in `arch/arm64/boot/dts/rockchip/rk3399-evb.dtsi`

```
/* Here "dfi" status must be "okay", it is due to lagacy code, the dmc node
is restriced by the dfi node. If the "dfi" node is disabled, it will also
invalidate the dmc node. So it is best to keep the status of the "dfi" node
consistent with dmc */
&dfi {
    status = "okay";
};

&dmc {
status = "okay";
......
system-status-freq = <
    /*system status         freq(KHz)*/
    SYS_STATUS_NORMAL       666000
    /* Remove the rest scenario */
    /*
    SYS_STATUS_REBOOT       528000
    SYS_STATUS_SUSPEND      200000
    SYS_STATUS_VIDEO_1080P  200000
    SYS_STATUS_VIDEO_4K     600000
    SYS_STATUS_VIDEO_4K_10B 800000
    SYS_STATUS_PERFORMANCE  800000
    SYS_STATUS_BOOST        400000
    SYS_STATUS_DUALVIEW     600000
    SYS_STATUS_ISP          600000
    */
>;
......
auto-min-freq = <666000>;
/* The value of "auto-freq-en" shall be 0 to disable load DDR Frequency
scaling */
auto-freq-en = <0>;
};
```

- For kernel 3.10, you need to find the node `clk_ddr_dvfs_table`, for example, RK3288 SDK's `clk_ddr_dvfs_table` is in `arch/arm/boot/dts/rk3288-tb_8846.dts`.

```
&clk_ddr_dvfs_table {
    operating-points = <
        /* KHz    uV */
```

```
        /* step 3,if the target frequency exceeds the maximun of this table,you
shall add the voltage table corresponding to the target frequency */
        200000 1050000
        300000 1050000
        400000 1100000
        533000 1150000
        >;

    freq-table = <
        /*status        freq(KHz)*/
        /* step 2, Comment out the other scenario,keep "SYS_STATUS_NORMAL" and
define it to you target frequency, for example you need 400MHz as below */
        SYS_STATUS_NORMAL   400000
        /*
        SYS_STATUS_SUSPEND  200000
        SYS_STATUS_VIDEO_1080P  240000
        SYS_STATUS_VIDEO_4K     400000
        SYS_STATUS_VIDEO_4K_60FPS     400000
        SYS_STATUS_PERFORMANCE  528000
        SYS_STATUS_DUALVIEW 400000
        SYS_STATUS_BOOST    324000
        SYS_STATUS_ISP      400000
        */
        >;
    bd-freq-table = <
        /* bandwidth   freq */
        5000            800000
        3500            456000
        2600            396000
        2000            324000
    >;
    auto-freq-table = <
        240000
        324000
        396000
        528000
        >;
    /* setp 1, set 0 to disable load DDR Frequency scaling */
    auto-freq=<0>;
    /*
     * 0: use standard flow
     * 1: vop dclk never divided
     * 2: vop dclk always divided
     */
    vop-dclk-mode = <0>;
    status="okay";
};
```

Just 3 steps can finish fixing frequency firmware.

1. The load frequency part should be set to 0
2. Comment out the other scenario,keep "SYS_STATUS_NORMAL" and define it to your target frequency
3. If the target frequency exceeds the maximun of this table,you shall add the voltage table corresponding to the target frequency.

- For kernel 3.0,you need to modify `dvfs_ddr_table` in `borad-**.c`. For example ,RK3066 SDK's `borad-**.c` is in `arch/arm/mach-rk30/board-rk30-sdk.c`

```
static struct cpufreq_frequency_table dvfs_ddr_table[] = {
    /*  */
    /* step 1. Comment out the other scene with leaving "DDR_FREQ_NORMAL" only
*/
    //{.frequency = 200 * 1000 + DDR_FREQ_SUSPEND, .index = 1050 * 1000},
    //{.frequency = 300 * 1000 + DDR_FREQ_VIDEO, .index = 1050 * 1000},
    /* step 2, Define "DDR_FREQ_NORMAL" to your target frequency,meanwhile pay
attention to whether the voltage match the frequency or not */
    {.frequency = 400 * 1000 + DDR_FREQ_NORMAL, .index = 1125 * 1000},
    {.frequency = CPUFREQ_TABLE_END},
};
```

Just 2 steps can finish fixing frequency firmware.

1. Comment out the other scene with leaving "DDR_FREQ_NORMAL" only
2. Define "DDR_FREQ_NORMAL" to your target frequency,meanwhile pay attention to whether the voltage match the frequency or not

How to get the DDR Bandwidth Utilization

Kernel 4.4 provides a command that can show the whole DDR bandwidth utilization,

```
rk3288:/sys/class/devfreq/dmc # cat load
11@396000000Hz
```

"11" Indicates that the current bandwidth utilization of DDR is 11%.

How to Test the Reliability of DDR

Please see the document "DDR-Verification-Process"

How to Check the Maximum Working Frequency of DDR

1. Add the frequency-voltage table to the corresponding frequency first, if you don't know how to ,please see the chapter "How to Modify DDR Frequency" and "How to Modify the Voltage Corresponding to A Certain DDR Frequency" .
2. Run google stressapptest from high frequency to low frequency, when you get an error, lower the frequency and run it again. No error, you can run it for more time. If it still works well, go to the next step.

"Google stressapptest" can be found in the file "DDR Verification Process" , which consists of introduction and software.We don't talk anymore here.

3. The previous step has roughly figured out the highest frequency.Now run a memtester .The same, when you get an error, lower the frequency and run it again. No error, you can run for a while, or no error, you can confirm the highest frequency point.

"memtester" can be found in the file "DDR Verification Process" , which consists of introduction and software.We don't talk anymore here.

"Google stressapptest" is a rough process,which can quickly report error. And " memtester" is more careful, so it reports error more slow. But “memtester” is mainly for the signal test, can cover the part that "google stressapptest" missing.

Apparently , the methods above are all based on the software test, which used to quickly get the maximum frequency. It is not sure the actual DDR SI can meet the JEDEC standard at the maximum frequency,that is necessary to measure the signal and burn-test.

How to Judge DDR in Self-Refresh Mode

It can be judged by measuring the CKE signals and it does not need an oscilloscope with a very high bandwidth.

CKE State	Explanation
Low level (Time>7.8us)	in self-refresh state
High level	in normal state

If the measured CKE is low period and high period, it is also can be regard as to the table above, that is, it enters the self-refresh mode and exit to normal state after a while.

Note: The time when CKE is low must be more than 7.8 us before self-refresh entry because power-down state also has a low CKE, but the time is less than 7.8 us. Please do not confuse it.

How to Judge DDR in Auto power-down Mode

It can be judged by measuring the CKE signals and it does not need an oscilloscope with a very high bandwidth.

CKE State	Explanation
Low level (Time<7.8us)	in power-down state
High level	in normal state

In the auto power-down mode, the measured CKE state holds low for nearly 7.8us (DDR3/DDR4) or 3.9us (LPDDR2/LPDDR3/ LPDDR4) and high for a short period of time, then enters low level for 7.8us or 3.9us for loop.

Note: The time when CKE is low must be less than 7.8 us(DDR3/DDR4) , 3.9us(LPDDR2/LPDDR3/LPDDR4) ,which can be judged a auto power-down.

How to Adjust the De-skew of DQ/DQS/CA/CLK

Mainly due to the unequal length of DDR routing in hardware PCB, the skew can be adjusted to achieve the effect similar to the same length of DDR routing. The skew function is the delay units in series on the signal line inside the DDR PHY. The delay of each signal line can be changed by controlling the number of delay units in series on each signal line through the skew register.

Adjusting the de-skew in kernel

Only RK322Xh/RK3328 support modifying the de-skew in kernel. The method is modify dts.

Chip Type：**RK322xh、RK3328**

Code location:

`arch/arm64/boot/dts/rk322xh-dram-default-timing.dtsi`

`arch/arm64/boot/dts/rk322xh-dram-2layer-timing.dtsi`

If customer have new file replace above file, please modify your new file.

Modify method:

According to the results of the released tool "deskew automatic scanning tool", select the `"mid"` value and add it to the corresponding dts definition.

Please according to "3228H deskew automatic scanning tool instruction. pdf" to use "deskew automatic scanning tool".

Adjusting the de-skew in loader

Only RK3308 support modifying the de-skew in loader.

Chip Type：**RK3308**

Required documents:

deskew automatic scanning tool, 3308_deskew.exe, RK3308_DDRXPXXXXXX_Template_VXX_de-skew.txt, rk3308_ddr_XXXMHz_uartX_mX_vX.XX.bin

Modify method:

According to the results of the released tool "deskew automatic scanning tool", select the `"mid"` value and add it to the corresponding definition in RK3308_DDRXPXXXXXX_Template_VXX_de-skew.txt. Using 3308_deskew.exe, change the definition of de-skew on rk3308_ddrxpxxxxxx_template_vxx_de-skew.txt to rk3308_ddr_xxxmhz_uartx_mx_vx.xx.bin.

Please according to "deskew automatic scanning tool instruction. pdf" to use "deskew automatic scanning tool".

Selection of RV1109/RV1126/RK356x DDR Frequency

For the RV1109/RV1126/RK356x platform loader, the frequency is changed 4 times, and the training result of the corresponding frequency is saved. The 4 frequency value allocated by the DMC in the kernel need to be consistent with the frequency pin the loader. Three of the default frequency in the RV1126/RV1109 loader are 328M, 528M, and 784M. The highest frequency point is reflected in the DDR bin name, such as rv1126_ddr_924MHz_v1.05.bin, and the last frequency point is 924M. Three of the default frequency points in RK356x loader are 324M, 528M, and 780M. The highest frequency point is also reflected in the ddr bin name, such as rk3568_ddr_1560MHz_v1.04.bin, and the last frequency point is 1560M. In addition, the frequency points in the loader can also be viewed through the serial port log. In the serial port log, the change to: xxxMHz is repeated 4 times, which represents the information of the 4 frequency points included. The frequency of the loader can also be modified through tools/ddrbin_tool in the rkbin directory. For specific usage rules, please refer to tools/ddrbin_tool_user_guide.txt.

For example, RK3568, if the highest frequency needs to be changed to 1332M, you need to change the ddr bin pointed to by Path1 and FlashData in RKBOOT/RK3568MINIALL.ini to 1332M in the rkbin project directory, and the frequency point in dts should be changed to 324M. 528M, 780M, 1332M, which matched your require. The actual operating frequency can only be one of these four.

Selection of RK3326S/PX30S DDR Frequency

For the RK3326S/PX30S platform, only 4 frequency points are supported. They are configed by ddrx_params node in `arch/arm64/boot/dts/rockchip/px30s-dram-default-timing.dtsi` and px30s_dmc_opp_table node in `arch/arm64/boot/dts/rockchip/px30.dtsi`.

For the ddrx_params node, if DDR3 for example, should config the freq_0, freq_1, freq_2, freq_3 in ddr3_params. If DDR4, should config the freq_0, freq_1, freq_2, freq_3 in ddr4_params.

For the px30s_dmc_opp_table node, all types of DDR are share a single frequency table. The enabling frequency must correspond to the freq_0, freq_1, freq_2, freq_3 in ddrx_params. For example, if LPDDR4, the enabled frequency of px30s_dmc_opp_table must correspond to freq_0, freq_1, freq_2, freq_3 in lpddr4_params. The configuration is as follows:

```
    px30s_dmc_opp_table: px30s-dmc-opp-table {
        compatible = "operating-points-v2";

        opp-328000000 {
            opp-hz = /bits/ 64 <328000000>;
            opp-microvolt = <1000000>;
        };
        opp-666000000 {
            opp-hz = /bits/ 64 <666000000>;
            opp-microvolt = <1000000>;
        };
        opp-786000000 {
            opp-hz = /bits/ 64 <786000000>;
            opp-microvolt = <1000000>;
        };
        opp-924000000 {
            opp-hz = /bits/ 64 <924000000>;
            opp-microvolt = <1000000>;
        };
        /* 1056M only for LP4 */
        opp-1056000000 {
            opp-hz = /bits/ 64 <1056000000>;
            opp-microvolt = <1000000>;
            status = "disabled";
        };
    };
```

```
/ {
    ...
    lpddr4_params: lpddr4-params {
        ...
        /* freq info, freq_0 is final frequency, unit: MHz */
        freq_0 = <924>;
        freq_1 = <328>;
        freq_2 = <666>;
        freq_3 = <786>;
        ...
    };
};
```

If the LPDDR4 needs to run 1056 MHz, you need to change one of the frequency values of freq_0, freq_1, freq_2, freq_3 in lpddr4_params to 1056. In addition, the px30s_dmc_opp_table node should disable old frequency points and add 1056000000 frequency points.

The new configuration is as follows:

```
    px30s_dmc_opp_table: px30s-dmc-opp-table {
        compatible = "operating-points-v2";

        opp-328000000 {
            opp-hz = /bits/ 64 <328000000>;
            opp-microvolt = <1000000>;
        };
        opp-666000000 {
            opp-hz = /bits/ 64 <666000000>;
            opp-microvolt = <1000000>;
        };
        opp-786000000 {
            opp-hz = /bits/ 64 <786000000>;
            opp-microvolt = <1000000>;
        };
        opp-924000000 {
            opp-hz = /bits/ 64 <924000000>;
            opp-microvolt = <1000000>;
            status = "disabled";
        };
        /* 1056M only for LP4 */
        opp-1056000000 {
            opp-hz = /bits/ 64 <1056000000>;
            opp-microvolt = <1000000>;
        };
    };
```

```
/ {
    ...
    lpddr4_params: lpddr4-params {
        ...
        /* freq info, freq_0 is final frequency, unit: MHz */
        freq_0 = <1056>;
        freq_1 = <328>;
        freq_2 = <666>;
        freq_3 = <786>;
        ...
    };
};
```

Enable RK3568 ECC

RK3568 supports ECC, if DDR ECC DQ0-7 has connected component, the loader will automatically enable ECC function. It should be noted that the DRAM on the ECC byte need to have the same row/bank/col as the component on the DQ0-31.