DDR Develop Guide

ID: RK-KF-YF-40

Release Version: V1.8.0

Release Date: 2024-05-17

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
2023.5.25	V1.5.0	Zhihuan He	Added the statement of “How to get the DDR manufacturer ID”
2023.5.31	V1.6.0	Zhihuan He	Added LPDDR5 Manufacturer ID table
2024.5.10	V1.7.0	CanYang He	Update content to sync with CH01-Rockchip_Developer_Guide_DDR_FAQ_CN.md
2024.5.17	V1.8.0	CanYang He	Update content to sync with CH01-Rockchip_Developer_Guide_DDR_FAQ_CN.md

Contents

1. 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.
```

In versions after kernel 3.10, DDR capacity information is no longer printed in the kernel.

2. How to Integrate Rockchip 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
PX30	`rk\rkbin\bin\rk33\px30_ddr_333MHz_vX.XX.bin`	
PX3SE	`rk\rkbin\bin\rk31\px3se_ddr_300MHz_vX.XX_uartX.bin`	
RK1808	`rk\rkbin\bin\rk1x\rk1808_ddr_XXXMHz_vX.XX.bin`	
RK3036	`rk\rkbin\bin\rk30\rk3036_ddr_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_XXXMHz_vX.XX.bin`	
RK322xh	`rk\rkbin\bin\rk33\rk322xh_ddr_333MHz_vX.XX.bin`	
RK3308	`rk\rkbin\bin\rk33\rk3308_ddr_XXXMHz_uartX_mX_vX.XX.bin`	
RK3326	`rk\rkbin\bin\rk33\rk3326_ddr_333MHz_vX.XX.bin`	
RK3328	`rk\rkbin\bin\rk33\rk3328_ddr_XXXMHz_vX.XX.bin`	
RK3368	`rk\rkbin\bin\rk33\rk3368_ddr_600MHz_vX.XX.bin`	
RK3399	`rk\rkbin\bin\rk33\rk3399_ddr_XXXMHz_vX.XX.bin`	2
RK3399PRO	`rk\rkbin\bin\rk33\rk3399pro_ddr_XXXMHz_vX.XX.bin`	
RK3528	`rk\rkbin\bin\rk35\rk3528_ddr_XXXMHz_vX.XX.bin`	3
RK3562	`rk\rkbin\bin\rk35\rk3562_ddr_XXXMHz_vX.XX.bin`	
RK3566	`rk\rkbin\bin\rk35\rk3566_ddr_XXXMHz_vX.XX.bin`	
RK3568	`rk\rkbin\bin\rk35\rk3568_ddr_XXXMHz_vX.XX.bin`	
RK3576	`rk\rkbin\bin\rk35\rk3576_ddr_lp4_XXXXMHz_lp5_XXXXMHz_vX.XX.bin`	
RK3588	`rk\rkbin\bin\rk35\rk3588_ddr_lp4_XXXXMHz_lp5_XXXXMHz_vX.XX.bin`	
RV1106	`rk\rkbin\bin\rv11\rv1106_ddr_XXXMHz_vX.XX.bin`	
RV1108	`rk\rkbin\bin\rv11\rv1108_ddr_vX.XX.bin`	
RV1126	`rk\rkbin\bin\rv11\rv1126_ddr_XXXMHz_vX.XX.bin`	
Future New Platform	They are all placed in the `rk\rkbin\bin` directory according to a similar naming convention, and can be searched for on their own	

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: RK3528 hardware design for PCB non-2-layer, DDR non-4BIT. For 2-layer PCB or 4BIT DDR, please refer to the "Special instructions for DDR bin" below.

Note 4: New platforms are placed in the `rk\rkbin\bin\` directory according to similar naming rules and can be searched by yourself.

3. Platform with 4 frequency points DDR bin

The DDR bin files of the following platforms all contain 4 DDR frequencies. 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.

Only these 4 frequencies can be used in the kernel. If you want to modify 4 frequencies, see the "Modify the DDR bin file" chapter.

Generally, the default 4 frequencies are as follows:

Platform	File	Contains frequency(MHz)
RK3528	`rk3528_ddr_XXXMHz_vX.XX.bin`	324,528,780,and the frequency in the file name
RK3562	`rk3562_ddr_XXXMHz_vX.XX.bin`	324,528,780,and the frequency in the file name
RK3566	`rk3566_ddr_XXXMHz_vX.XX.bin`	324,528,780,and the frequency in the file name
RK3568	`rk3568_ddr_XXXMHz_vX.XX.bin`	324,528,780,and the frequency in the file name
RK3576	`rk3576_ddr_lp4_XXXXMHz_lp5_XXXXMHz_vX.XX.bin`	LP4/LP4X:528,1068,1560,and the frequency in the file name LP5/LP5X:534,1320,1968,and the frequency in the file name
RK3588	`rk3588_ddr_lp4_XXXXMHz_lp5_XXXXMHz_vX.XX.bin`	LP4/LP4X:528,1068,1560,and the frequency in the file name LP5/LP5X:534,1320,1968,and the frequency in the file name
RV1126	`rv1126_ddr_XXXMHz_vX.XX.bin`	328,528,784,and the frequency in the file name

These 4 frequency points can be viewed from the loader's serial port log, as follows,

```
DDR ... v1.14
LPDDR4X, 2112MHz
channel[0] BW=16 Col=10 Bk=8 CS0 Row=16/0 CS1 Row=16/0 CS=2 Die BW=16 Size=2048MB
channel[1] BW=16 Col=10 Bk=8 CS0 Row=16/0 CS1 Row=16/0 CS=2 Die BW=16 Size=2048MB
channel[2] BW=16 Col=10 Bk=8 CS0 Row=16/0 CS1 Row=16/0 CS=2 Die BW=16 Size=2048MB
channel[3] BW=16 Col=10 Bk=8 CS0 Row=16/0 CS1 Row=16/0 CS=2 Die BW=16 Size=2048MB
...
change to F1: 528MHz  /* 4 DDR frequencies */
change to F2: 1068MHz
change to F3: 1560MHz
change to F0: 2112MHz
out
```

You can also use the `rkbin/tools/ddrbin_tool` to read the current four frequencies, such as

`./ddrbin_tool px30 -g gen_param.txt px30_ddr_333MHz_v1.15.bin`

For more about ddrbin_tool, refer to detailed instructions.

4. Special instructions for DDR bin

For the DDR bin files under rk\rkbin\bin, there are some special that need to be explained.

- RK3399

Because different DDR types of RK3399 support different frequencies, and the DDR bin name does not specify the frequency by type, it causes confusion for some customers.The following is a explanation.

File	**DDR3/LP3 frequency**		**LP4 frequency**	
	Frequency of exiting loader(MHz)	Frequency supported by kernel (MHz)	Frequency of exiting loader(MHz)	Frequency supported by kernel (MHz)
`rk33\rk3399_ddr_666MHz_v1.30.bin`	666	<=666	416	328, 416, 666, 856
`rk33\rk3399_ddr_800MHz_v1.30.bin`	800	<=800	856	328,416, 666, 856
`rk33\rk3399_ddr_933MHz_v1.30.bin`	933	<=933	856	328, 416, 666, 856, 933

- bin with "eyescan" suffix

The DDR bin suffixed with eyescan is used to synthesize a Loader that can obtain the 2D eye diagram of the DDR signal, such as

`rk\rkbin\bin\rk35\rk3566_ddr_1056MHz_eyescan_v1.16.bin`

`rk\rkbin\bin\rk35\rk3588_ddr_lp4_2112MHz_lp5_2736MHz_eyescan_v1.11.bin`

`rk\rkbin\bin\rk35\rk3562_ddr_1332MHz_eyescan_v1.04.bin`

`rk\rkbin\bin\rk35\rk3568_ddr_1560MHz_eyescan_v1.16.bin`

- bin with "ultra" suffix

ultra low power for e-books, used with e-book hardware, such as

`rk\rkbin\bin\rk35\rk3566_ddr_XXXMHz_ultra_v1.10.bin`

`rk\rkbin\bin\rk35\rk3562_ddr_XXXMHz_ultra_v1.05.bin`

- bin with "tb" suffix

It is used for quick boot. When using it, you must select the bin corresponding to the DDR type, such as

`rk\rkbin\bin\rv11\rv1126_tpl_XXXMHz_ddr4_tb_v1.08.bin`

`rk\rkbin\bin\rv11\rv1126_tpl_XXXMHz_ddr3_tb_v1.08.bin`

`rk\rkbin\bin\rv11\rv1126_tpl_XXXMHz_lp3_tb_v1.08.bin`

`rk\rkbin\bin\rv11\rv1126_tpl_XXXMHz_lp4_tb_v1.08.bin`

Used for RV1106 quick boot, used for DDR3

`rk\rkbin\bin\rv11\rv1106_ddr_924MHz_tb_v1.13.bin`

- 3528 with "PCB"

For 4BIT DDR design

`rk\rkbin\bin\rk35\rk3528_ddr_1056MHz_4BIT_PCB_v1.07.bin`

For 2-layer PCB design

`rk\rkbin\bin\rk35\rk3528_ddr_1056MHz_2L_PCB_v1.07.bin`

5. Modify the DDR bin file

The DDR bin file mentioned in the above chapter is used to initialize DDR when booting. Through the tools provided by Rockchip, you can modify the DDR bin file to modify DDR initialization parameters, DDR frequency, close the serial port, change serial port baud rate, etc.

- rk_ddrBin_tool_windows

It is recommended to use rk_ddrBin_tool_windows, which is a tool with a user interface and is easy to use.

Tools are available at:

https://redmine.rock-chips.com/documents/49 -> rk_ddrBin_tool_windows_Vx.xx.7z

Generally, there are several compressed packages,and you need to download them all and then decompress them.

Under the Help of the tool, there is a usage guide.

- `rkbin/tools/ddrbin_tool`

This tool is in command line mode. Under the rkbin project, `rkbin/tools/ddrbin_tool`

The documentation is `rkbin/tools/ddrbin_tool_user_guide.txt`

Frequently seen modifications:

- Change DDR frequency

For SOCs that only support 4 DDR frequencies, see the chapter "Platform with 4 frequency points DDR bin".

You can modify the corresponding 4 frequencies according to the DDR type. For example, below shows the four frequencies of LPDDR4

	参数名称	配置值	有效值	Unit	
1	lp4_freq	2112	300-2133	MHz	DDR初始化频率(FSP_0频率)
2	lp4_f1_freq_mhz	528	300-2133	MHz	DDR FSP_1频率，用于变频
3	lp4_f2_freq_mhz	1068	300-2133	MHz	DDR FSP_2频率，用于变频
4	lp4_f3_freq_mhz	1560	300-2133	MHz	DDR FSP_3频率，用于变频

- Change serial port

参数名称	配置值	有效值	Unit	
uart id	0		——	串口ID，0xf=关闭串口打印

- Change serial port baud rate

参数名称	配置值	有效值	Unit	
uart id	0	|	——	串口ID，0xf=关闭串口打印
uart iomux	0	|	——	串口IOMUX
uart baudrate	1500000	|	bps	串口波特率，支持115200或1500000

6. How to Change DDR Frequency in U-Boot

- RK322x

The following modification method is only supported by RK322x. 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.

- RK3576/RK3588

The following modification method is only supported by RK3576 and RK3588. When the loader initializes DDR, these platforms will initialize 4 DDR frequencies together for subsequent kernel use, and will exit the loader at the highest frequency by default and continue with the following procedures.

The following F1, F2, F3, and F0 are these four frequencies. The default is to exit with F0.

```
DDR ... v1.14
LPDDR4X, 2112MHz
channel[0] BW=16 Col=10 Bk=8 CS0 Row=16/0 CS1 Row=16/0 CS=2 Die BW=16 Size=2048MB
channel[1] BW=16 Col=10 Bk=8 CS0 Row=16/0 CS1 Row=16/0 CS=2 Die BW=16 Size=2048MB
channel[2] BW=16 Col=10 Bk=8 CS0 Row=16/0 CS1 Row=16/0 CS=2 Die BW=16 Size=2048MB
channel[3] BW=16 Col=10 Bk=8 CS0 Row=16/0 CS1 Row=16/0 CS=2 Die BW=16 Size=2048MB
...
change to F1: 528MHz  /* 4个DDR频率 */
change to F2: 1068MHz
change to F3: 1560MHz
change to F0: 2112MHz
out
```

When you need to modify the DDR frequency under U-Boot, you only need to modify the frequency of exiting the loader.

Through the tool described in "Modify the DDR bin file", find the `boot_fsp` parameter, and you can choose which frequency of F0/F1/F2/F3 is used as the frequency to exit the loader. In this way, the DDR frequency in U-Boot can be modified.

boot_fsp	0	|	——	完成DDR初始化后进入系统的DDR频率(F0/F1/F2/F3)。

7. 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 and later versions, 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 1: For RK3576 and RK3588, after enabling DDR scaling function through the dts **dmc** node, check whether the "center-supply" and "mem-supply"attribute under the **dmc** node is correctly configured based on the power connect used by the actual product hardware. The configuration values of these attributes represent the name of the regulator nodes that provide power to the DDR module on the SOC-side as well as to the LOGIC.

  The default for RK3576 is `center-supply = <&vdd_ddr_s0>`,`mem-supply = <&vdd_logic_s0>`.

  The default for RK3588 is `center-supply = <&vdd_ddr_s0>`,`mem-supply = <&vdd_log_s0>`.

  If the **dmc** node lacks this attribute, the **dmc** driver will fail to load. The kernel log is as follows.

  ```
  rockchip-dmc dmc: Cannot get the regulator "center"
  ```

  Note 2: For other platforms, after enabling DDR scaling function through the dts **dmc** node, check whether the "center-supply" attribute under the **dmc** node is correctly configured based on the power connect used by the actual product hardware. The configuration value of this attribute represent the name of the regulator node that provide power to the DDR module on the SOC-side.

  The default for RK3399 is `center-supply = <&vdd_center>`;

  and the default for other platforms such as PX30/RK3568/RV1126 is `center-supply=<&vdd_logic>`.

  If the **dmc** node lacks this attribute, the **dmc** driver will fail to load. The kernel log is as follows.

  ```
  rockchip-dmc dmc: Cannot get the regulator "center"
  ```

  Note 3: 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 {
    center-supply = <&vdd_center>; /* This requires the customer to configure it
according to the actual hardware circuit*/
    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},
};
```

8. 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 and later versions

  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);
```

9. How to Check the DDR Capacity

If you look for a DDR capacity roughly, using the following command to check the MemTotal capacity. This capacity 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 printing of DDR capacity information in kernel 4.4 and later versions;Some chip have DDR capacity information 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/ kernel 4.4 and later versions
RK3026	√	√
RK3028A	√	√
RK3036	√	×
RK3066	√	√
RK3126	√	√
RK3126B, RK3126C with trust.img	√	×
RK3126B, RK3126C without trust.img	√	√
RK3128	√	√
RK3188	√	√
RK322x	√	×
RK322xh	√	×
RK3288	√	√
RK3328	√	×
RK3368	√	×
RK3399	√	×
RV1108	√	×
Other SOC	√	×

√ 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" or "Total capacity" when SOC chip only has 1 DDR channel or the sum of two channel's "Size" or "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
```

10. Modify DDR capacity

At present, the DDR capacity of all Rockchip platforms is automatically recognized and does not require customers to configure DDR capacity. Methods for modifying DDR capacity are provided here, mainly for customers to evaluate performance or evaluate the impact of reducing DDR capacity.

In "Rockchip_Developer_Guide_UBoot_Nextdev_CN" -> "CH08 Debugging Methods" -> "Modify DDR Capacity", another modification method is provided.

When booting, the DDR initialization code will pass the DDR capacity to U-Boot, and U-Boot will remove some secure memory and then pass it to the kernel. Users can modify the DDR capacity passed to the kernel during the U-Boot stage.

Code location:

```
./arch/arm/mach-rockchip/param.c
```

Modification steps:

1. Add print to view current DDR capacity information

```
struct memblock *param_parse_ddr_mem(int *out_count)
{
    ......
    for (i = 0, n = 0; i < count; i++, n++) {
        base = t->u.ddr_mem.bank[i];
        size = t->u.ddr_mem.bank[i + count];
        printf("base:0x%llx, size:0x%llx\n", base, size);    //Add this line to
print

        /* 0~4GB */
        if (base < SZ_4GB) {
            mem[n].base = base;
            mem[n].size = ddr_mem_get_usable_size(base, size);
    ......
}
```

After making the above modifications, recompile U-Boot and download it. In the new U-Boot startup log, there is the following output.

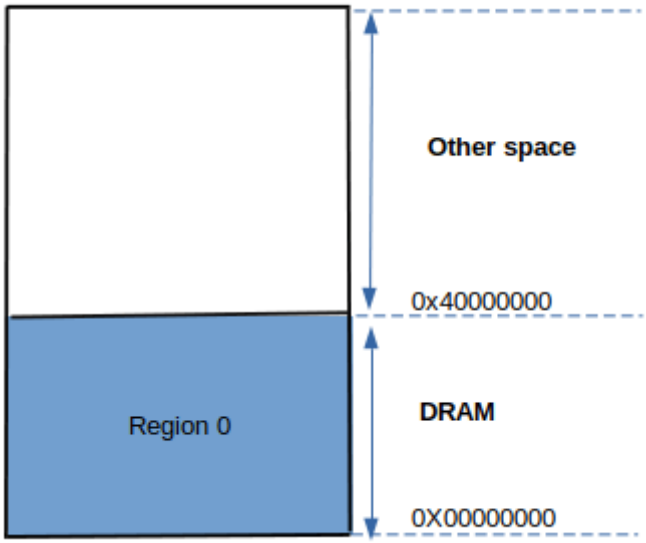
```
U-Boot 2017.09-g0236dc3682-220712-dirty #hcy (Aug 04 2022 - 10:46:07 +0800)

Model: Rockchip RK3568 Evaluation Board
PreSerial: 2, raw, 0xfe660000
DRAM:  base:0x0, size:0x40000000   //This line is the output we added
```

in

```
base:0x0, size:0x40000000
```

As shown in the figure below, there is 1 DRAM space, the base address is 0x0, and the size is 0x40000000



Before U-Boot boots the kernel, U-Boot removes some secure memory and passes the DDR capacity information to the kernel as follows.

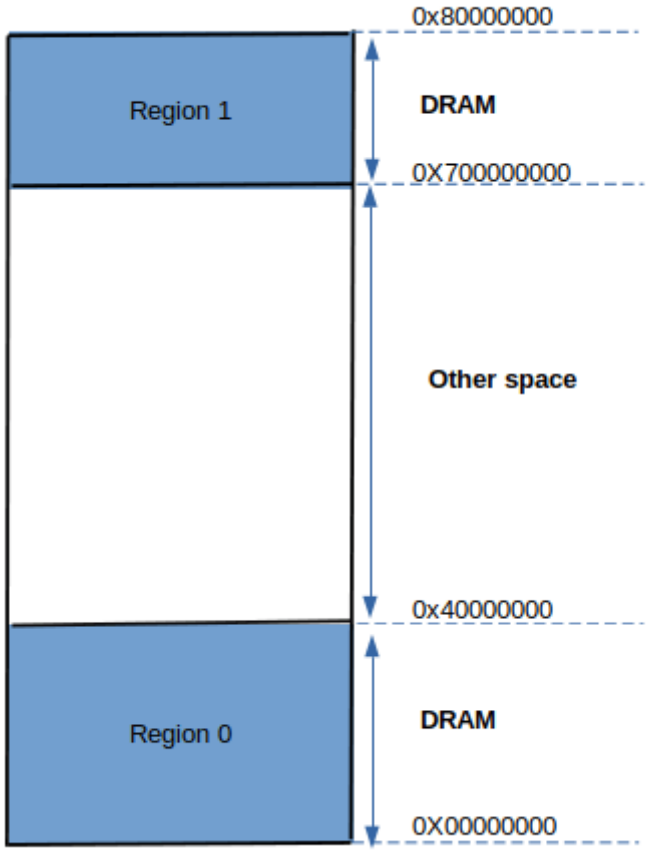
```
Adding bank: 0x00200000 - 0x08400000 (size: 0x08200000)  //After removing the
secure memory block
Adding bank: 0x09400000 - 0x40000000 (size: 0x36c00000)  //After removing the
secure memory block
Total: 1182.666 ms

Starting kernel ...

[    0.000000] Booting Linux on physical CPU 0x0000000000 [0x412fd050]
```

2. After understanding the description method of the above DDR capacity information, you can modify it.

Assume that a piece of DRAM space is added to the system, as shown in the location of Region 1 in the figure below. You can make the following modifications:




```
struct memblock *param_parse_ddr_mem(int *out_count)
{
    ......
    /* extend top ram size */
    if (t->u.ddr_mem.flags & DDR_MEM_FLG_EXT_TOP)
        gd->ram_top_ext_size = t->u.ddr_mem.data[0];

    //Modify DDR capacity information,starting here
    count = 2;
    t->u.ddr_mem.count = count;
    t->u.ddr_mem.bank[0] = 0x0;  //Region 0 base address
    t->u.ddr_mem.bank[0 + count] = 0x40000000 - 0x0; //Region 0 size
    t->u.ddr_mem.bank[1] = 0x70000000;  //Region 1 base address
    t->u.ddr_mem.bank[1 + count] = 0x80000000 - 0x70000000;  //Region 1 size
    t->u.ddr_mem.hash = 0;
    ///Modify DDR capacity information, end here

    /* normal ram size */
    count = t->u.ddr_mem.count;
    mem = calloc(count + MEM_RESV_COUNT, sizeof(*mem));
    if (!mem) {
        printf("Calloc ddr memory failed\n");
        return 0;
    }
    ......
    for (i = 0, n = 0; i < count; i++, n++) {
        base = t->u.ddr_mem.bank[i];
        size = t->u.ddr_mem.bank[i + count];
        printf("base:0x%llx, size:0x%llx\n", base, size);
```

```
        /* 0~4GB */
        if (base < SZ_4GB) {
            mem[n].base = base;
            mem[n].size = ddr_mem_get_usable_size(base, size);
    ......
}
```

Let's focus on analyzing the following code:

```
    count = 2;    //Indicates that there are 2 blocks of DRAM space
    t->u.ddr_mem.count = count;    //Must be written like this
    t->u.ddr_mem.bank[0] = 0x0;
    t->u.ddr_mem.bank[0 + count] = 0x40000000 - 0x0; //The array index must have
"+count"
    //Region base address plus Region size is One group, how many groups there
are, must be the same as the "count" above
    t->u.ddr_mem.bank[1] = 0x70000000;
    t->u.ddr_mem.bank[1 + count] = 0x80000000 - 0x70000000;  //The array index
must have "+count"
    t->u.ddr_mem.hash = 0;   //Must be written like this
```

Recompile U-Boot and download it. In the new boot log, you can see that the modifications have taken effect.

```
......
U-Boot 2017.09-g0236dc3682-220712-dirty #hcy (Aug 04 2022 - 10:46:07 +0800)

Model: Rockchip RK3568 Evaluation Board
PreSerial: 2, raw, 0xfe660000
DRAM:  base:0x0, size:0x40000000
base:0x70000000, size:0x10000000   //this is what we have added
......
Adding bank: 0x00200000 - 0x08400000 (size: 0x08200000)
Adding bank: 0x09400000 - 0x40000000 (size: 0x36c00000)
Adding bank: 0x70000000 - 0x80000000 (size: 0x10000000)  //this is what we have
added
Total: 1182.666 ms

Starting kernel ...

[    0.000000] Booting Linux on physical CPU 0x0000000000 [0x412fd050]
```

If the customer still only has one DRAM, but wants to change the size to 0x80000000, he can add the following code.

```
    count = 1;    //Indicates that there is 1 block
    t->u.ddr_mem.count = count;    //Must be written like this
    t->u.ddr_mem.bank[0] = 0x0;
    t->u.ddr_mem.bank[0 + count] = 0x80000000 - 0x0; //The array index must have
"+count"
    t->u.ddr_mem.hash = 0;   //Must be written like this
```

==Note: The DRAM capacity cannot be increased at will, because without real storage, the system will be abnormal. Generally, the size is changed to a smaller size for evaluation. Region1 is added to the example here for ease of understanding.==

11. Check DDR frequency

- by devfreq

If the DDR frequency scaling function is enabled, the current DDR frequency can be obtained through the following nodes

`cat /sys/class/devfreq/dmc/cur_freq`

```
console:/ # cat /sys/class/devfreq/dmc/cur_freq
780000000
```

- by clk_summary

`cat /sys/kernel/debug/clk/clk_summary | grep scmi_clk_ddr`

or

`cat /sys/kernel/debug/clk/clk_summary | grep sclk_ddrc`

or

`cat /sys/kernel/debug/clk/clk_summary | grep sclk_ddr`

Depends on the kernel version.

```
console:/ # cat /sys/kernel/debug/clk/clk_summary | grep scmi_clk_ddr
scmi_clk_ddr                          0        0        0  2736000000          0
  0  50000 /*2736000000 is the DDR frequency in Hz, that is, 2736MHz*/
```

12. 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 later versions) and kernel 3.10 has some difference.

kernel 4.4 and later versions:

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.

- PX30S/RK3326S, see "PX30S/RK3326S DDR frequency selection" on the back
- The platforms listed in Chapter "Platform with 4 frequency points DDR bin", see "Platform with 4 frequency points DDR bin, its DDR frequency selection" on the back
- For kernel 4.4, it requires get the **dmc** node in dts. For example, **dmc** node in RK3399 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 1==: 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.

==Note 2==: For RK3399 LPDDR4 and RV1126 platforms, the frequency supported by DDR frequency scaling has been determined in the loader stage, and these frequencies will be printed out through the serial port during the DDR initialization stage, and the kernel frequencies defined by dmc_opp_table needs to correspond to it.

Taking RV1126 as an example, the DDR frequencies printed in the loader stage are 328MHz, 528MHz, 784MHz, and 924MHz, so dmc_opp_table can only define these four frequencies.

```
change to: 328MHz
change to: 528MHz
change to: 784MHz
change to: 924MHz(final freq)
```

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

    opp-328000000 {
        opp-hz = /bits/ 64 <328000000>;
        opp-microvolt = <800000>;
    };
    opp-528000000 {
        opp-hz = /bits/ 64 <528000000>;
        opp-microvolt = <800000>;
    };
    opp-784000000 {
        opp-hz = /bits/ 64 <784000000>;
        opp-microvolt = <800000>;
    };
    opp-924000000 {
        opp-hz = /bits/ 64 <924000000>;
        opp-microvolt = <800000>;
    };
};
```

If the frequencies define by dmc_opp_table and the loader are inconsistent, a frequency mismatch problem will occur when the kernel dmc driver performs DDR frequency scaling. The log is as follows.

```
rockchip-dmc dmc: Get wrong frequency, Request 1056000000, Current 924000000
```

- 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 ".

12.1 RK3399 LPDDR4 supports 928MHz modification method

1. Modify the RKBOOT/RK3399MINIALL.ini file in the rkbin directory, select the 933MHz DDR bin file, and pack the loader.

```
diff --git a/RKBOOT/RK3399MINIALL.ini b/RKBOOT/RK3399MINIALL.ini
index d8e71dd7..3e20f255 100755
--- a/RKBOOT/RK3399MINIALL.ini
+++ b/RKBOOT/RK3399MINIALL.ini
@@ -5,7 +5,7 @@ MAJOR=1
 MINOR=26
```

```
 [CODE471_OPTION]
 NUM=1
-Path1=bin/rk33/rk3399_ddr_800MHz_v1.27.bin
+Path1=bin/rk33/rk3399_ddr_933MHz_v1.27.bin
 Sleep=1
 [CODE472_OPTION]
 NUM=1
@@ -14,7 +14,7 @@ Path1=bin/rk33/rk3399_usbplug_v1.26.bin
 NUM=2
 LOADER1=FlashData
 LOADER2=FlashBoot
-FlashData=bin/rk33/rk3399_ddr_800MHz_v1.27.bin
+FlashData=bin/rk33/rk3399_ddr_933MHz_v1.27.bin
 FlashBoot=bin/rk33/rk3399_miniloader_v1.26.bin
 [OUTPUT]
 PATH=rk3399_loader_v1.27.126.bin
```

After download the pack loader file into the machine, you can see from the log printed by the serial port that the loader supports five frequency scaling: 416MHz, 856MHz, 328MHz, 666MHz, and 928MHz. The current running DDR frequency is 856MHz.

```
DDR Version 1.27 20211018
In
......
support 416 856 328 666 928 MHz, current 856MHz
OUT
```

2. Modify the kernel dts dmc_opp_table node and change the `status = "disabled"` in the opp-928000000 table to `status ="okay"`, or delete it directly.

```
                opp-928000000 {
                        opp-hz = /bits/ 64 <928000000>;
                        opp-microvolt = <900000>;
-                       status = "disabled";
                };
```

3. Modify the kernel dts dmc node and change all "856000" in the system-status-freq table to "928000", or refer to this document The "How to Modify DDR Frequency" chapter describes the SYS_STATUS_XXX related Scenario frequency scaling, which can be selectively modified according to the DDR bandwidth requirements of different scenarios in the actual project.

12.2 PX30S/RK3326S DDR frequency selection

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>;
        ...
    };
};
```

12.3 Platform with 4 frequency points DDR bin, its DDR frequency selection

The platforms listed in Chapter "Platform with 4 frequency points DDR bin", excluding RV1126, all other platforms are compatible with the methods describe in this section.

These platforms' loader changes the frequency 4 times and saves the training results of the corresponding frequencies.

The following F1, F2, F3, and F0 are these four frequencies. The default is to exit with F0.

```
DDR ... v1.14
LPDDR4X, 2112MHz
channel[0] BW=16 Col=10 Bk=8 CS0 Row=16/0 CS1 Row=16/0 CS=2 Die BW=16 Size=2048MB
channel[1] BW=16 Col=10 Bk=8 CS0 Row=16/0 CS1 Row=16/0 CS=2 Die BW=16 Size=2048MB
channel[2] BW=16 Col=10 Bk=8 CS0 Row=16/0 CS1 Row=16/0 CS=2 Die BW=16 Size=2048MB
channel[3] BW=16 Col=10 Bk=8 CS0 Row=16/0 CS1 Row=16/0 CS=2 Die BW=16 Size=2048MB
...
change to F1: 528MHz  /* 4 DDR frequencies*/
change to F2: 1068MHz
change to F3: 1560MHz
change to F0: 2112MHz
out
```

For these platforms, although the frequency supported by DDR frequency scaling is also determined in the loader stage, the kernel dmc driver will automatically obtain the supported DDR frequency value, so there is no dmc_opp_table frequency matching problem.
The kernel dmc nodes for these platforms are as follows

```
dmc: dmc {
    ...
    system-status-freq = <
    /*system status          freq(KHz)*/
    SYS_STATUS_NORMAL        DMC_FREQ_LEVEL_MID_HIGH
    SYS_STATUS_REBOOT        DMC_FREQ_LEVEL_MID_LOW
    ...
    SYS_STATUS_SUSPEND       DMC_FREQ_LEVEL_LOW
    ...
    SYS_STATUS_PERFORMANCE   DMC_FREQ_LEVEL_HIGH
    ...
    >;
    ...
};
```

The frequencies corresponding to each scene are no longer direct frequencies, but definitions such as DMC_FREQ_LEVEL_HIGH. These definitions can automatically match to the 4 frequencies of the loader. If you use the "Modify the DDR bin file" method to modify 4 frequencies, DMC_FREQ_LEVEL_HIGH and other definitions can automatically change to the modified 4 frequencies.
Taking the above four frequencies of the boot loader as an example, the corresponding relationship is as follows:

Macro definition	**frequency**	**illustrate**
DMC_FREQ_LEVEL_HIGH	F0 2112MHz	Highest frequency
DMC_FREQ_LEVEL_MID_HIGH	F3 1560MHz	sub-high frequency
DMC_FREQ_LEVEL_MID_LOW	F2 1068MHz	sub-low frequency
DMC_FREQ_LEVEL_LOW	F1 528MHz	lowest frequency

13. 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:
Versions after kernel 4.4: You can directly adjust the voltage of a certain regulator through the `/sys/kernel/debug/regulator/` node. DDR frequency must be fixed first, otherwise, the regulate voltage here will conflict with the regulate voltage of DDR frequency scaling. DDR fixed frequency reference chapter "How to fix DDR frequency".

Take 3588 regulate voltage as an example:

```
cat /sys/kernel/debug/regulator/vdd_ddr_s0/voltage
echo 700000 > /sys/kernel/debug/regulator/vdd_ddr_s0/voltage
```

The regulators that need to be adjusted to modify the voltage corresponding to the DDR frequency on each platform are as follows:

Platform	**Nodes that need to be adjusted**
RK3308/RK3308B	"vdd_core" or "vdd_log"
RK3576	"vdd_logic_s0" and "vdd_ddr_s0"
RK3588	"vdd_log_s0" and "vdd_ddr_s0"
Other SOC	"vdd_center" or "vdd_logic"

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`

  Take RK3399 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-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 `borad-**.c` 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.

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

- For kernel 4.4 and later versions, 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.

15. How to Fix DDR Frequency

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

kernel 4.4 and later versions:

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 and later versions, 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

16. How to get the DDR Bandwidth Utilization

Kernel 4.4 and later versions 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%.

Rockchip also provides an executable file dedicated to observing DDR bandwidth. Please refer to the "Rockchip DDR Bandwidth Tool Instructions" chapter for details.

17. How to Test the Reliability of DDR

Please see the document "DDR-Verification-Process"

18. 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. See the "Rockchip DDR DQ Eye Tool Guide" chapter to confirm the eye diagram for platforms that provide eye diagram tools.
3. 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 in
https://redmine.rock-chips.com/documents/49 -> DDR related information_VerX.XX.7z -> DDR particle verification_DDR test resource file-> static_stressapptest
For details on how to use it, see the "Guide_DDR_Verification_Process" chapter.

4. 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, If it still works well, you can confirm the highest frequency point.

memtester is at
https://redmine.rock-chips.com/documents/49 -> DDR related information_VerX.XX.7z -> DDR particle verification_DDR test resource file-> static_memtester
For details on how to use it, please refer to the "Guide_DDR_Verification_Process" chapter.

"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.

19. 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.

20. 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.

21. 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.

21.1 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".

21.2 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".

22. Run DDR stress test under U-Boot

Under U-Boot, we have porting two commonly used DDR stress test programs, stressapptest and memtester.

Mainly used for:

- When the kernel fails to start, perform a stress test on the DDR under U-Boot to eliminate DDR problems.
- The kernel does not have the conditions to run stressapptest or memtester.

  For example, if there are many incomplete modules in the kernel, it will cause system instability and make stressapptest or memtester unable to run for a long time.
- Test larger DDR space

  The kernel can only test the remaining DDR space due to system usage. Testing under U-Boot occupies much less space and can test a larger DDR space.
- Specify physical address test

22.1 stressapptest

enable

`make ARCH=arm64 menuconfig ->Command line interface -> DDR Tool -> Enable DDR Tool`

and enable

`make ARCH=arm64 menuconfig ->Command line interface -> DDR Tool -> Enable DDR Tool -> Enable stressapptest for ddr`

Compile U-Boot and download uboot.img.

Restart the device. On the PC serial terminal, keep pressing the `CTRL+C` keys until U-Boot enters the command line mode, as follows

```
  pclk_top_root 100000 KHz
  aclk_low_top_root 396000 KHz
Net:   No ethernet found.
Hit key to stop autoboot('CTRL+C'):  0
=> <INTERRUPT>
=> <INTERRUPT>
```

Enter `stressapptest -h`, and follow the parameter instructions to test.

22.2 memtester

enable

`make ARCH=arm64 menuconfig ->Command line interface -> DDR Tool -> Enable DDR Tool`

and enable

`make ARCH=arm64 menuconfig ->Command line interface -> DDR Tool -> Enable DDR Tool -> Enable memtester for ddr`

Compile U-Boot and download uboot.img.

Restart the device. On the PC serial terminal, keep pressing the `CTRL+C` keys until U-Boot enters the command line mode, as follows

```
  pclk_top_root 100000 KHz
  aclk_low_top_root 396000 KHz
Net:   No ethernet found.
Hit key to stop autoboot('CTRL+C'):  0
=> <INTERRUPT>
=> <INTERRUPT>
```

Enter `memtester -h`, and follow the parameter instructions to test.

23. Enable RK3568 ECC

ECC refers to Error Correcting Code, and DDR ECC performs error checking and correction on DDR data.

RK3568 supports SideBand ECC, which adds a DDR die next to the DDR data channel dedicated to storing ECC data.

Its ECC has the ability to correct 1 bit and detect 2 bit errors, namely SEC/DED ECC (Single Error Correction/Double Error Detection).

1. Enable ECC: As long as DDR_ECC_DQ0-7 has DDR chip attached, ECC will be automatically enabled.
2. The function of DDR_ECC_DQ0-7: The implementation of ECC is 32-bit DQ data + 7-bit ECC data. DDR_ECC_DQ0-7 is used to store the ECC data calculated from DQ0-DQ31. Therefore, an additional DDR chip needs to be attached in DDR_ECC_DQ0-7 to store ECC data.
3. Requirements for the additional DDR chip: the DRAM type; number of rows, columns, banks; all must be the same as the DDR chip on DQ0-31.
4. Supported DRAM types: All DRAM types support ECC.

24. How to obtain DDR ECC information in the kernel

DDR ECC information is in the edac architecture of the kernel. For platforms that support DDR ECC function, such as RK3568, and the hardware have DDR ECC enabled(see the "Enable RK3568 ECC" chapter for details). To obtain DDR ECC information, first enable the edac module in dtsi. If the platform dtsi does not have an edac node, it means that the platform does not support it.

```
    edac: edac {
        ...
        status = "okay";
    };
```

RK EDAC will be registered to the system at the following location:

```
sys/devices/system/edac/mc/mc0
or
sys/bus/edac/devices/mc/mc0/
```

Get RK EDAC name:

```
# cat sys/devices/system/edac/mc/mc0/mc_name
rk_edac_ecc
```

Get the number of single-bit correctable errors (ce) accumulated after this startup:

```
# cat sys/devices/system/edac/mc/mc0/ce_count
0
```

Get the number of multi-bit uncorrectable errors (ue) accumulated after this startup:

```
# cat sys/devices/system/edac/mc/mc0/ue_count
0
```

When the system restarts, the kernel will reinitialize the related EDAC modules and counters, that is, the number of previously recorded errors will be cleared and counting will start from zero.

25. How to get the DDR manufacturer ID

Only the LPDDR types (such as LPDDR2, LPDDR3, LPDDR4, LPDDR4X, LPDDR5, LPDDR5X) have manufacturer IDs, while the DDR types (such as DDR2, DDR3, DDR4) do not have manufacturer IDs.

25.1 dmcdbg node through kernel

The following platforms support obtaining DDR manufacturer ID through `proc/dmcdbg/dmcinfo`

- RK3566
- RK3568

- RK3326
- RK3326S

You need to first enable the dmcdbg node on the corresponding platform dtsi (if the node does not exist, this platform does not support it). For example, RK356X, in the `arch/arm64/boot/dts/rockchip/rk3568.dtsi` file, enable the dmcdbg node:

```
dmcdbg: dmcdbg {
    compatible = "rockchip,rk3568-dmcdbg";
    status = "okay";
};
```

After recompiling, update the kernel firmware, and booting it. You can get the DDR manufacturer ID and other DDR information through the command, cat `proc/dmcdbg/dmcinfo`.

```
console:/ # cat proc/dmcdbg/dmcinfo
DramType:       LPDDR4
Dram ID:        MR5=0x1,MR6=0x0,MR7=0x1 /* The MR5 is the manufacturer ID, and
MR6 and MR7 are the version IDs reserved for the manufacturer definition */

DramCapacity:
CS Count:       1
Bus Width:      32 bit
Column:         10
Bank:           8
CS0_Row:        16
CS1_Row:        0
DieBusWidth:    16 bit
TotalSize:      2048 MB

devfreq/dmc:    Enable
governor:       dmc_ondemand

cur_freq:       780000000

NOTE:
more information about dmc can get from /sys/class/devfreq/dmc.
```

25.2 Output information through loader

RK3576/RK3588 loader output contains manufacturer ID

```
DDR ... v1.14
LPDDR5, 2736MHz
channel[0] BW=16 Col=10 Bk=16 CS0 Row=15 CS1 Row=15 CS=2 Die BW=16 Size=2048MB
channel[1] BW=16 Col=10 Bk=16 CS0 Row=15 CS1 Row=15 CS=2 Die BW=16 Size=2048MB
channel[2] BW=16 Col=10 Bk=16 CS0 Row=15 CS1 Row=15 CS=2 Die BW=16 Size=2048MB
channel[3] BW=16 Col=10 Bk=16 CS0 Row=15 CS1 Row=15 CS=2 Die BW=16 Size=2048MB
Manufacturer ID:0xff /* This is the manufacturer ID */
...
change to F0: 2736MHz
out
```

RK3399 loader output contains manufacturer ID

```
DDR Version ...
In
channel 0
...
MR5=0xFF    /* This is the manufacturer ID */
...
change freq to 416MHz 0,1
Channel 0: LPDDR4,416MHz
Bus Width=32 Col=10 Bank=8 Row=16 CS=1 Die Bus-Width=16 Size=2048MB
Channel 1: LPDDR4,416MHz
Bus Width=32 Col=10 Bank=8 Row=16 CS=1 Die Bus-Width=16 Size=2048MB
...
OUT
```

25.3 Points to Note

The following points should be noted:

1. Only LPDDR types (such as LPDDR2, LPDDR3, LPDDR4, LPDDR4X, LPDDR5, LPDDR5X) have manufacturer ID, DDR type (such as DDR2, DDR3, DDR4) does not have a manufacturer ID.
2. The manufacturer ID shows the manufacturer of the wafer, not the packaged DDR brand information. Because many DDR vendors do not have the ability to produce wafers.
3. For all LPDDR, the model number on the packaging screen print cannot be obtained. This is different from Nand Flash or eMMC.
4. The resolution of the wafer manufacturer ID follows JEP166 of JEDEC standard.

25.4 Manufacturer ID table

The following is the latest Manufacturer ID table as of the completion of this document.

There will be updates in the future. You can search JEP166 at https://www.jedec.org to find the latest Manufacturer ID table.

LPDDR2, LPDDR3 Manufacturer ID table:

MR5 value	Manufacturer name
0x1	Samsung
0x2	Qimonda
0x3	Elpida
0x4	Etron
0x5	Nanya
0x6	SK hynix
0x7	Mosel
0x8	Winbond
0x9	ESMT
0xa	Zentel
0xb	Spansion
0xc	SST
0xd	ZMOS
0xe	Intel
0x12	Being Advanced Memory Corp
0x1a	Xi'an UniIC Semiconductors Co., Ltd
0x1b	ISSI
0x1c	JSC
0xaa	Tezzaron
0xc2	Macronix
0xf8	Fidelix
0xfc	eveRAM
0xfd	AP Memory
0xfe	Numonyx
0xff	Micron

LPDDR4 Manufacturer ID table:

MR5 value	Manufacturer name
0x1	Samsung
0x5	Nanya
0x6	SK hynix
0x8	Winbond
0x9	ESMT
0x13	CXMT
0x1a	Xi'an UniIC Semiconductors Co., Ltd
0x1c	JSC
0xf8	Fidelix
0xf9	Ultra Memory
0xfd	AP Memory
0xff	Micron

LPDDR5 Manufacturer ID table:

MR5 value	Manufacturer name
0x1	Samsung
0x5	Nanya
0x6	SK hynix
0x8	Winbond
0x9	ESMT
0x13	CXMT
0xe5	Dosilicon
0xff	Micron

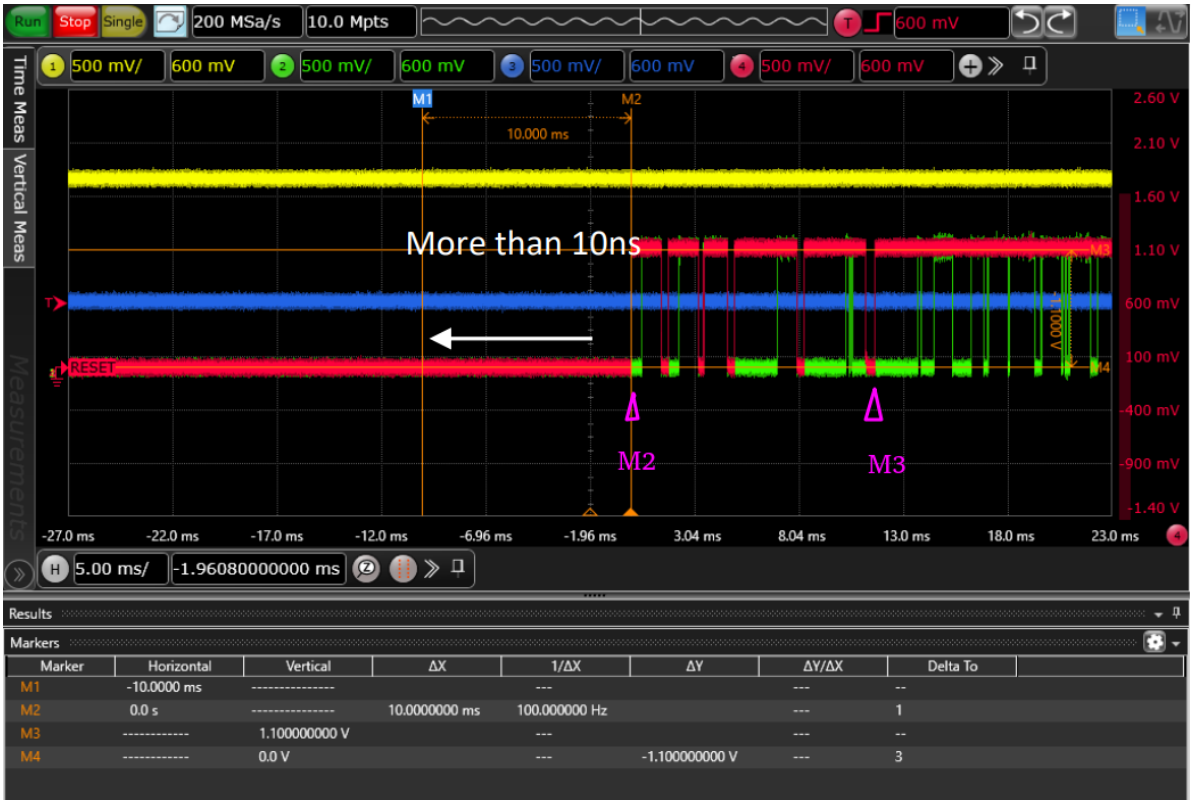
26. Signal Related FAQs

26.1 tINIT3 does not satisfy the protocol

Reason: When the Rockchip platform performs DDR initialization, the RESET signal will reset multiple times. and the measurement manufacturer mistakenly used the high level of the first RESET signal as the measurement point of tINIT3, so the measured tINIT3 does not meet the protocol.

For the correct measurement method, the last RESET signal high level should be used as the measurement point of tINIT3. As shown below,

M2 cannot be used as the measurement point, but M3 should be used as the measurement point.



26.2 Note of enable DDR ODT

- According to JEDEC standards, LPDDR4/LPDDR4X do not support enable ODT below 800MHz.
- According to JEDEC standards, DDR4 do not support enable ODT below 625MHz.

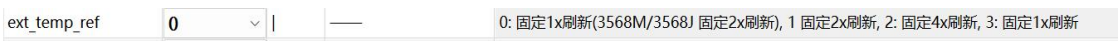
27. High temperature refresh rate problem

When the temperature is higher than 85°C, the DDR needs to be refreshed twice rate.

This requirement requires special attention for industrial and automotive standard products.

Rockchip platform has the following methods to double refresh rate:

- LPDDR3, LPDDR4, LPDDR4X, LPDDR5, and LPDDR5X have their own temperature sensors, which can tell the required refresh rate through MR4 based on the temperature. Some Rockchip platforms implement a set of Automatic Temperature Derating functions, referred to as derate. The refresh rate can be automatically adjusted based on the MR4 information provided by the DDR die. This enables refresh rate and temperature matching.
- For DDR3 and DDR4, since they do not have a temperature sensor and do not have the function of MR4 to tell the required refresh rate, derate cannot be used. This type of DDR die requires forced double refreshed rate.
- Customers can modify the ext_temp_ref of the DDR bin to double refresh rate



For modification method, see the "Modify the DDR bin file" chapter for details

Note: Rockchip's industrial chips, automotive chips, and wide-temperature chips all have refresh rates modified, and there is no need to double them by yourself through this tool.

In the tool, the corresponding relationship between fixed 2x/4x/1x refresh is as follows

Name	LPDDR3/LPDDR4	LPDDR5	Remark
Fixed 2x refresh	Refresh Rate in MR4=0x4	Refresh Multiplier in MR4=0xB	2x refresh
Fixed 4x refresh	Refresh Rate in MR4==0x5	Refresh Multiplier in MR4=0xC	4x refresh
Fixed 1x refresh	Refresh Rate in MR4==0x3	Refresh Multiplier in MR4=0x9	Normal refresh

The current Rockchip's industrial chips, automotive chips, and wide-temperature chips processing methods and DDR bin version requirements are as follows:

Platform	DDR type	Processing methods and DDR bin version requirements
RK3308J/M RK3308K RK3308GK	All DDR types	Use `rk3308_ddr_XXXMHz_uartX_mX_v1.26.bin` and later versions default is 4x refresh (i.e. 0.25x tREFI)
RK3568J/M	DDR3 DDR4	Use `rk3568_ddr_XXXXMHz_v1.16.bin` and later versions default is 2x refresh (i.e. 0.5x tREFI)
RK3568J/M	LPDDR3 LPDDR4 LPDDR4X	Use `rk3568_ddr_XXXXMHz_v1.18.bin` and later versions Enable the derate function by default
RK3588J/M	LPDDR4 LPDDR4X LPDDR5 LPDDR5X	Use `rk3588_ddr_XXXXMHz_v1.13.bin` and later versions Enable the derate function by default
RV1126K	All DDR types	Use `rv1126_ddr_XXXXMHz_v1.03.bin` and later versions default is 2x refresh (i.e. 0.5x tREFI)

The platforms with default DDR bin has the derate function enabled, as follows

Platform	DDR type	Processing methods and DDR bin version requirements
RK3528	LPDDR3 LPDDR4 LPDDR4X	Use `rk3528_ddr_XXXXMHz_v1.07.bin` or `rk3528_ddr_XXXXMHz_2L_PCB_v1.07.bin` or rk3528_ddr_XXXXMHz_4BIT_PCB_v1.07.bin` and later versions Enable the derate function by default
RK3562	LPDDR3 LPDDR4 LPDDR4X	Use `rk3562_ddr_XXXXMHz_v1.05.bin` or `rk3562_ddr_XXXXMHz_ultra_v1.05.bin` and later versions Enable the derate function by default
RK3568	LPDDR3 LPDDR4 LPDDR4X	Use `rk3568_ddr_XXXXMHz_v1.18.bin` and later versions Enable the derate function by default
RK3576	LPDDR4 LPDDR4X LPDDR5 LPDDR5X	Use `rk3576_ddr_XXXXMHz_v1.00.bin` and later versions Enable the derate function by default
RK3588	LPDDR4 LPDDR4X LPDDR5 LPDDR5X	Use `rk3588_ddr_XXXXMHz_v1.13.bin` and later versions Enable the derate function by default