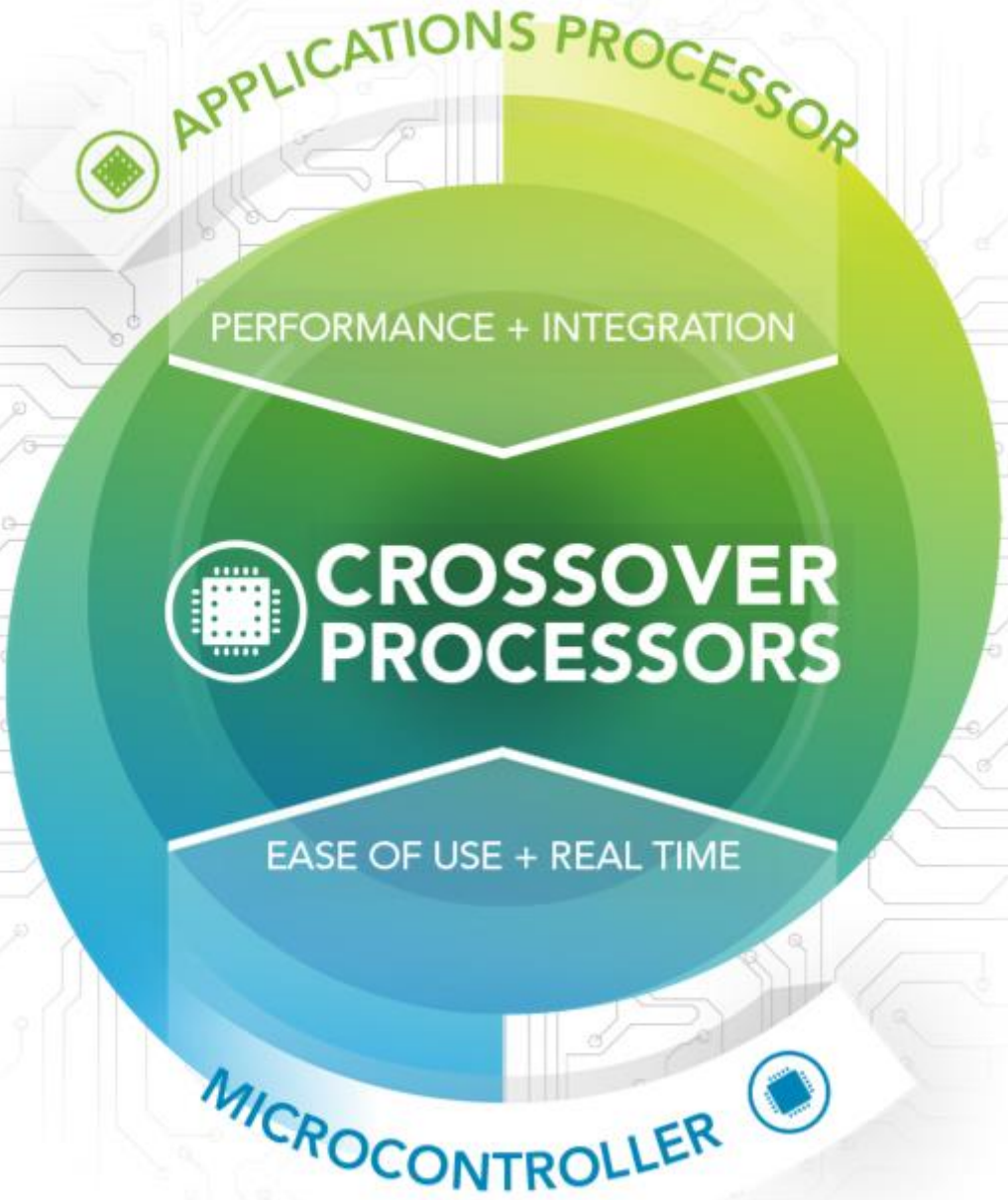
I.MX RT Design Tips and FAQ

JI CHENG /DAVID

2020.09



iMX RT Agenda

- iMXRT Design Tips
  - Hardware Design
  - Software Design
- iMXRT Learning Materials
- iMXRT FAQ

IMX RT Design Tips —Hardware

硬件设计checking list

本PPT主要目的在于：着重提炼用户设计过程中容易忽视的地方，并不能替代用户手册Reference Manual, 数据手册DataSheet，及硬件指导手册

《Hardware Development Guide for the MIMXRT1050/MIMXRT1060 Processor 》。

尤其硬件指导手册是硬件设计工程师必须要首先完成阅读的内容。

英文版本：

https://www.nxp.com/docs/en/user-guide/MIMXRT105060HDUG.pdf

中文版本：

https://www.nxp.com/docs/zh/user-guide/MIMXRT105060HDUG.pdf

RT的电源域管理

3. 电源

有关电源域和电源去耦建议，请参见表 1。请注意，功率控制图适用于 MIMXRT1050 A1 芯片。

表 1. 电源域

电源导轨	最小值 (V)	典型值 (V)	最大值 (V)	说明
VDD_SOC_IN	0.925	—	1.3	内核供电电源
VDD_HIGH_IN	2.8	3.3	3.6	VDD_HIGH_IN 电源电压
DCDC_IN	2.8[1]	2.9[1]	3.3[1]	DCDC 电源
VDD_SNVS_IN	2.4	3	3.6	SNVS 和 RTC 电源
USB_OTG1_VBUS USB_OTG2_VBUS	4.4	5	5.5	USB VBUS 电源
VDDA_ADC	3	3.3	3.6	12 位 ADC 电源
VDDA_IN	3	3.3	3.6	12 位 ADC 电源
NVCC_SD0	3	3.3	3.6	SDIO1 组中的 GPIO 电源（3.3 V 模式）
	1.65	1.8	1.95	SDIO1 组中 GPIO 电源（1.8 V 模式）
NVCC_SD1	3	3.3	3.6	SDIO2 组中 GPIO 电源（3.3 V 模式）
	1.65	1.8	1.95	SDIO2 组中 GPIO 电源（1.8 V 模式）
NVCC_EMC	3	3.3	3.6	EMC 组中 GPIO IO 电源（3.3 V 模式）
	1.65	1.8	1.95	EMC 组中 GPIO IO 电源（1.8 V 模式）
NVCC_GPIO	3	3.3	3.6	GPIO IO 电源

1. 对于 A0 芯片，DCDC_IN 电压域为 2.8 V-3.0 V，而对于 A1 版本及以后的芯片的 DCDC_IN，电压域为 2.8 V-3.6 V。

即使作为GPIO，供电域也是由NVCC_SD、EMC决定

Table 83. 10 x 10 mm functional contact assignments

Ball Name	10 x 10 Ball	Power Group	Ball Type	Default Setting			
				Default Mode	Default Function	Input/ Output	Value
CCM_CLK1_N	P13	—	—	—	CCM_CLK1_N	—	—
CCM_CLK1_P	N13	—	—	—	CCM_CLK1_P	—	—
GPIO_AD_B0_00	M14	NVCC_GPIO	Digital GPIO	ALT5	GPIO1.IO[0]	Input	Keeper
GPIO_AD_B0_01	H10	NVCC_GPIO	Digital GPIO	ALT5	GPIO1.IO[1]	Input	Keeper
GPIO_AD_B0_02	M11	NVCC_GPIO	Digital GPIO	ALT5	GPIO1.IO[2]	Input	Keeper
GPIO_AD_B0_03	G11	NVCC_GPIO	Digital GPIO	ALT5	GPIO1.IO[3]	Input	Keeper
GPIO_AD_B0_04	F11	NVCC_GPIO	Digital GPIO	ALT0	SRC.BOOT.MODE[0]	Input	100 K PD
GPIO_AD_B0_05	G14	NVCC_GPIO	Digital GPIO	ALT0	SRC.BOOT.MODE[1]	Input	100 K PD
GPIO_AD_B0_06	E14	NVCC_GPIO	Digital GPIO	ALT0	JTAG.MUX.TMS	Input	47 K PU
GPIO_AD_B0_07	F12	NVCC_GPIO	Digital GPIO	ALT0	JTAG.MUX.TCK	Input	47 K PU

Datasheet文档中Table83定义了管脚所属电源域以及每个管脚上电后默认的属性

RT的电源去耦设计

表 2. 电源去耦建议

电源导轨	去耦和大容量电容（最小量）	说明
VDD_SOC_IN	5 x 0.22 µF[2] + 1 x 4.7 µF[1] + 1 x 22 µF[3]	在 F6、F7、F8 引脚旁放置至少 1 个 4.7µF 电容和 3 个 0.22µF 电容。
VDD_HIGH_IN	1 x 0.22 µF[2] + 1 x 4.7 µF[1]	—
VDD_HIGH_CAP	1 X 0.22 µF[2] + 1 x 4.7 µF[1]	VDDHIGH_CAP 仅限 MX6RT 负载。
DCDC_IN	3 x 0.22 µF[2] + 1 x 4.7 µF[1] + 1 x 22 µF[3]	在 M1 引脚旁放置至少 1 个 4.7µF 电容和 1 个 0.22µF 电容。
VDD_SNVS_IN	1 x 0.22 µF[2]	—
VDD_SNVS_CAP	1 x 0.22 µF[2] + 1 x 4.7 µF[1]	选择低 ESR 小电容器。请勿将任何负载连接到 VDD_SNVS_CAP。
USB_OTG1_VBUS USB_OTG2_VBUS	1 x 1 µF[1]	额定值为 10-V。
VDDA_ADC_3P3	1 x 0.22 µF[2] + 1 x 1 µF[1]	将降压电压和旁路电容放在 N14 引脚旁。
NVCC_SD0	1 x 0.1 µF + 1 x 4.7 µF[1]	将降压电压和旁路电容放在 J6 引脚旁。
NVCC_SD1	1 x 0.1 µF + 1 x 4.7 µF[1]	将降压电压和旁路电容放在 K5 引脚旁。
NVCC_EMC	2 x 0.1 µF + 1 x 4.7 µF[1]	将降压电容和旁路电容放在 F5 和 E6 引脚旁。
NVCC_GPIO	3 x 0.1 µF + 1 x 4.7 µF[1]	—

1. 对于 4.7µF 电容，请使用 0402 封装。
2. 对于 0.22µF 电容，请使用 0402 封装。
3. 对于 22µF 电容器，0603 封装是首选；也可采用 0805 和 1206 封装。

Power Management Design

在设计RT最小系统的电源管理部分时，涉及到外部电源管理芯片LDO或者DCDC的选型，需要知悉RT芯片各个电源域的最大电流消耗以评估电源芯片的最大供电能力，下表可供用户评估。

Notes: VDD_SOC_IN的电流是由内部DCDC来提供，所以其来源于DCDC_IN，所以评估总电流时不需要把VDD_SOC_IN的消耗考虑在内，而IO电源域NVCC_GPIO，NVCC_SD以及NVCC_EMC的供电需求则由用户的实际外部接口电路来决定。

Table 11. Maximum supply currents

Power Rail	Conditions	Max Current	Unit
DCDC_IN	Max power for FF chip at 105 °C	100	mA
VDD_HIGH_IN	Include internal loading in analog	50	mA
VDD_SNVS_IN	—	250	µA
USB_OTG1_VBUS USB_OTG2_VBUS	25 mA for each active USB interface	50	mA
VDDA_ADC_3P3	3.3 V power supply for 12-bit ADC, 600 µA typical, 750 µA max, for each ADC. 100 Ohm max loading for touch panel, cause 33 mA current.	40	mA
NVCC_GPIO NVCC_SD0 NVCC_SD1 NVCC_EMC	Imax = N x C x V x (0.5 x F) Where: N—Number of IO pins supplied by the power line C—Equivalent external capacitive load V—IO voltage (0.5 x F)—Data change rate. Up to 0.5 of the clock rate (F) In this equation, Imax is in Amps, C in Farads, V in Volts, and F in Hertz.		

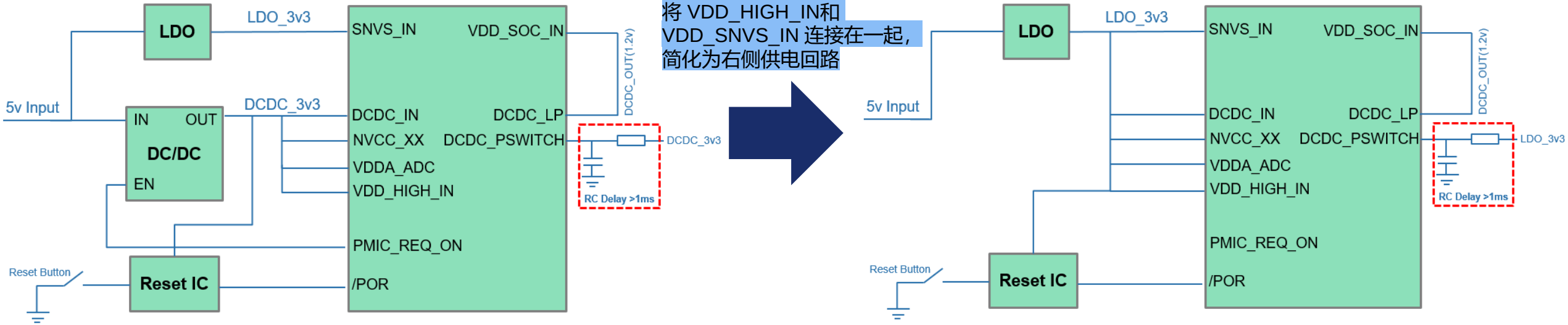
Power Rail	Overdrive (600MHz)			Full Speed Run (528MHz)			Low Speed Run (132MHz)			Low Power Run (24MHz)		
	Voltage (V)	Current (mA)	Power (mW)	Voltage (V)	Current (mA)	Power (mW)	Voltage (V)	Current (mA)	Power (mW)	Voltage (V)	Current (mA)	Power (mW)
DCDC_IN	3.3	75.3	248.49	3.3	56.5	186.5	3.3	13.70	45.21	3.3	2.26	7.4
VDD_HIGH_IN	3.3	19.840	65.5	3.3	19.280	63.6	3.3	10.060	33.2	3.3	0.310	1.02
VDD_SNVS_IN	3.3	0.068	0.224	3.3	0.055	0.18	3.3	0.024	0.08	3.3	0.015	0.050

RT的电源简化设计

1. In general, a critical power sequence should be followed on i.MXRT HW design. SNVS power domain should be POR ahead of other power domain just following below EVK schematic design, but if the lowest sleep power mode(SNVS Mode) is not necessary in your app, SNVS_IN can be POR together with other power domain to implement true one LDO or DC-DC regulator in system.

- VDD_SNVS_IN supply must be turned on before any other power supply or be connected (shorted) with VDD_HIGH_IN supply
- If a coin cell is used to power VDD_SNVS_IN, then ensure that it is connected before any other supply is switched on
- POR_B should be held low during the entire power up sequence
- Power Down Sequence Requirement:
  - VDD_SNVS_IN supply must be turned off after any other power supply or be connected (shorted) with VDD_HIGH_IN supply
  - If a coin cell is used to power VDD_SNVS_IN, then ensure that it is removed after any other supply is switched off
- Following power rails are fully independent, no power up/power down sequence requirement:
  - USB_OTG1_VBUS
  - USB_OTG2_VBUS
  - VDDA_ADC_3P3

如果在系统掉电期间不需要实时时钟功能/开关机功能等，则可将 VDD_HIGH_IN和 VDD_SNVS_IN 连接在一起，简化为右侧供电回路



PMIC REQ ON/PMIC_STBY_REQ信号应用

13.6 WAKEUP Pin

• This chip supports the use of a WAKEUP pin (GPIO5_IO00 ALT5) to request main SoC

• power on to exit SNVS mode. To use it, follow the tips below.

• SNVS must be in Dumb PMIC mode (*default*, and must for on-chip DCDC)

• Configure IOMUXC_SNVS to select WAKEUP pin ALT5 to mux it to GPIO5_IO00

• Configure GPIO5 ICR to either low level or high level sensitive

• Set GPIO5 IMR bit0 to enable GPIO5_IO00 interrupt

• Enter SNVS mode by SW or ON/OFF button, then on-chip DCDC will be off, and PMIC_ON_REQ pin will also go to low to notify outside

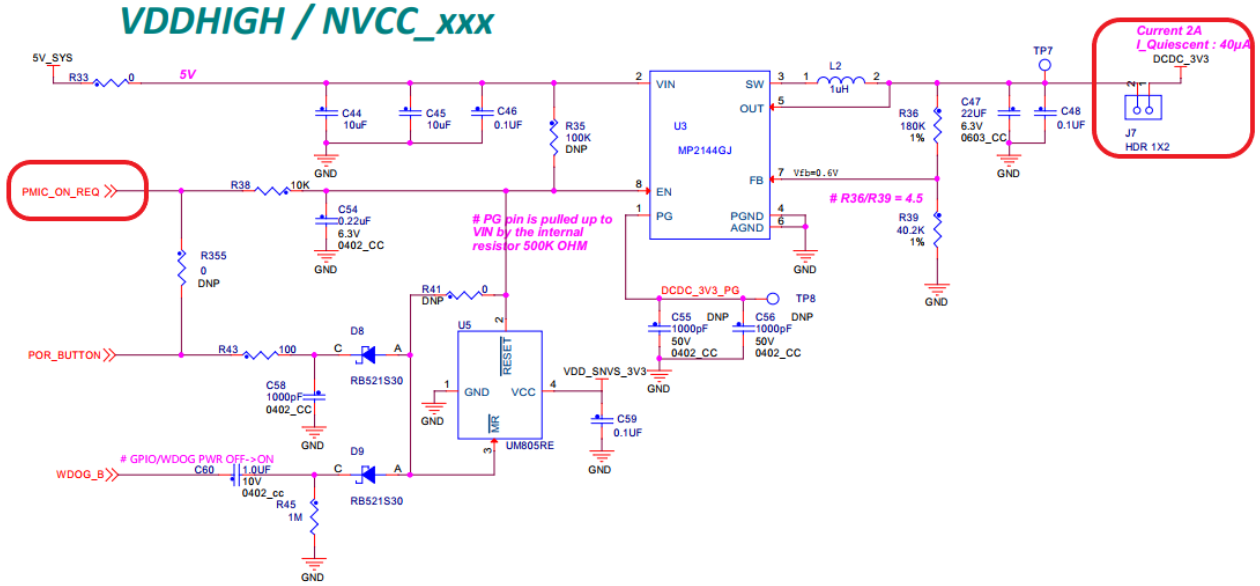
• Assert WAKEUP pin for at least two 32 KHz cycles (active high or low depends on GPIO5 ICR configuration), so that GPIO5 interrupt gets sampled by SNVS module

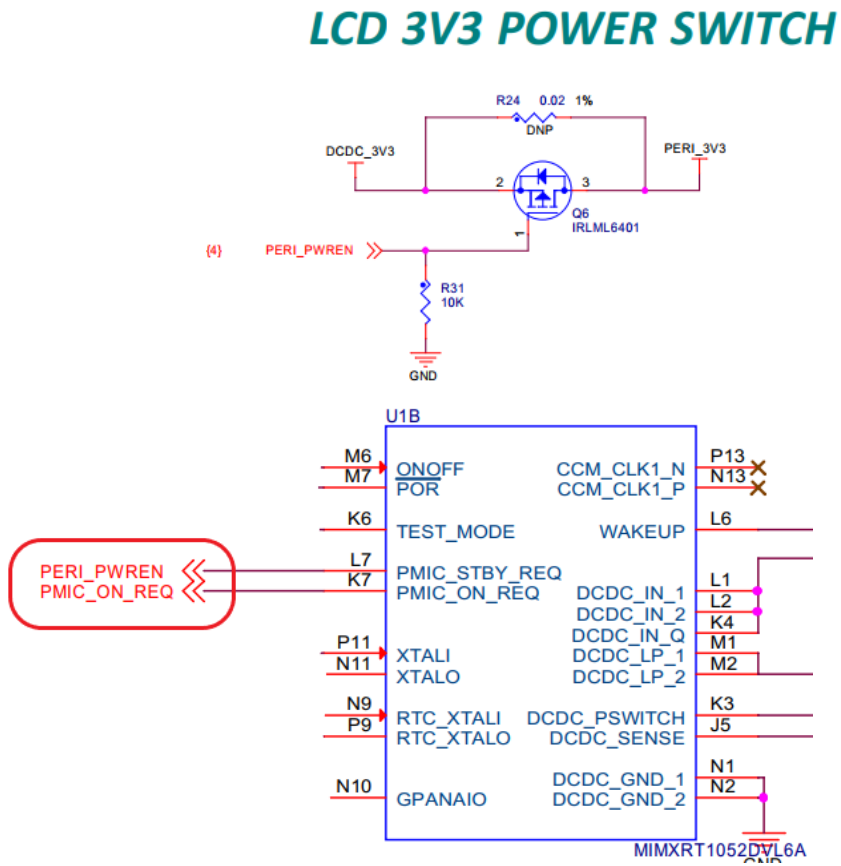
• SNVS will assert PMIC_ON_REQ pin high and on-chip DCDC begins to ramp on

• SoC power on procedure (except SNVS) begins (ROM boot, …)

1.PMIC_REQ_ON：在关机状态下为低，按开机键后或者Wakeup pin唤醒后，REQ_ON将输出高，以驱动使能外部DCDC

2. PMIC_STBY_REQ : RT进入normal run，该管脚为low，进入suspend模式，DSM mode，该管脚为high。该管脚用于系统进入低功耗模式的时候将外部设备关闭，比如在RT-EVK上使用该管脚控制外部LCD电源供电。进入snvs状态，该管脚也输出为low。





RT各电源域上下电时序要求

- 上电序列要求
  - VDD_SNVS_IN 电源必须在任何其他电源之前开启，或者与 VDD_HIGH_IN 电源一块上电
  - 如果使用纽扣电池为 VDD_SNVS_IN 供电，则确保在接通任何其他电源之前将其连通
  - 当内部 DCDC 使能时，需要外部延迟电路以在 DCDC_IN 稳定后使“DCDC_PSWITCH”信号延迟 1 ms 以上
  - 在整个上电序列期间，POR_B 应保持低电平
- 掉电序列要求
  - VDD_SNVS_IN 电源必须在任何其他电源之后关闭，或者与 VDD_HIGH_IN 电源连接（短路）
  - 如果使用纽扣电池为 VDD_SNVS_IN 供电，则确保在关断任何其他电源之后将其移除

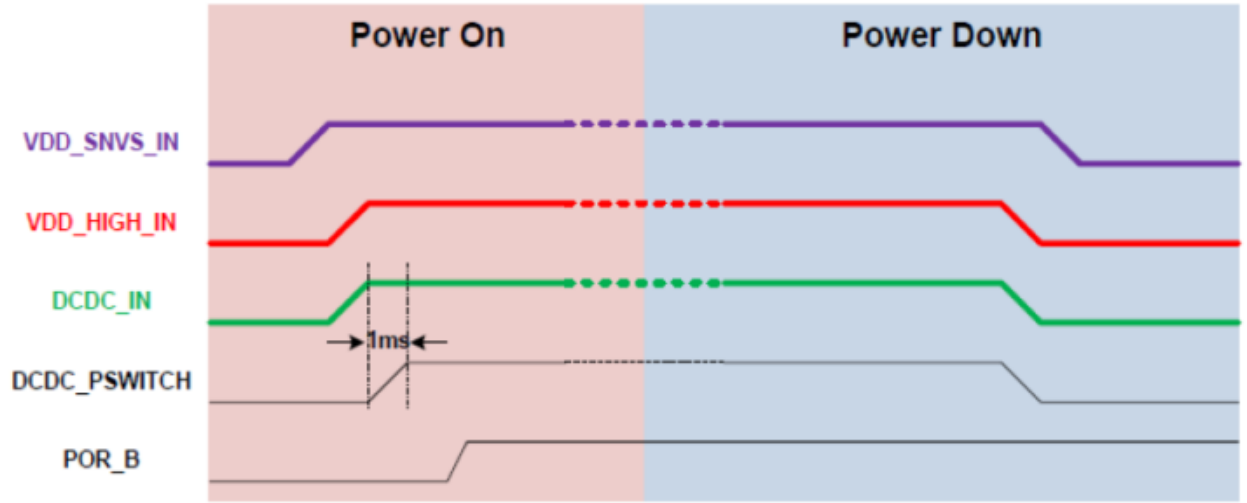
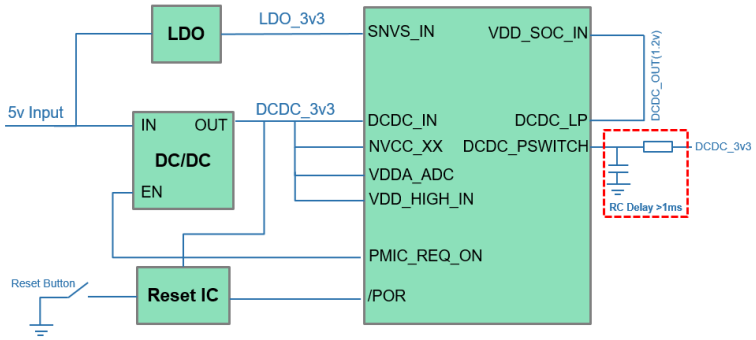


图 1. 上电和掉电序列

RT内部DCDC设计要点（一）

IMXRT has a DC-DC inside to generate 1.2v for VDD_SOC_IN to save BOM cost and to improve power efficiency. But to make this inside DC-DC module working normally a >1ms RC delay circuit should be added on DCDC_PSWITCH pin to make PSWITCH POR later than DCDC_IN, and surely don't forget the LC circuit on DCDC_LP pin to comprise the completed DCDC circuit;



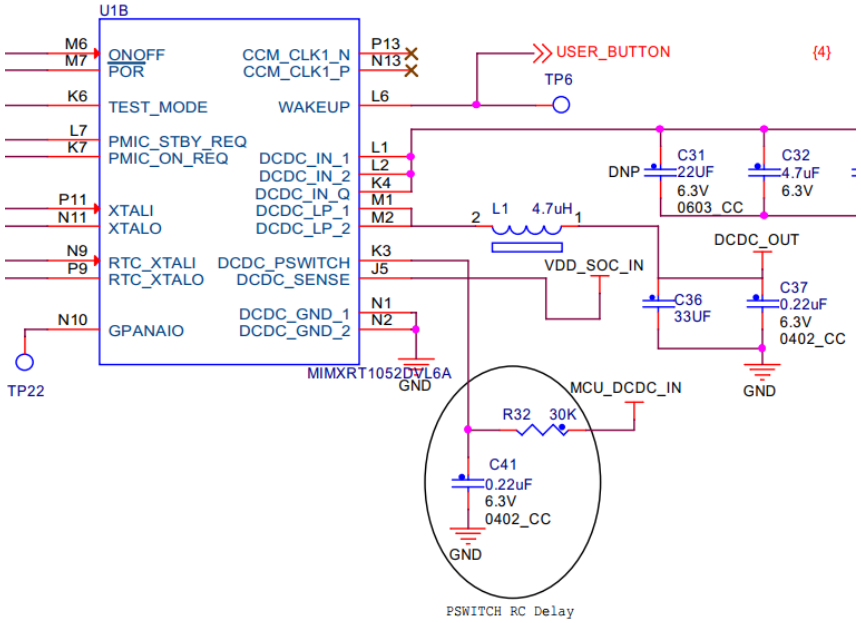
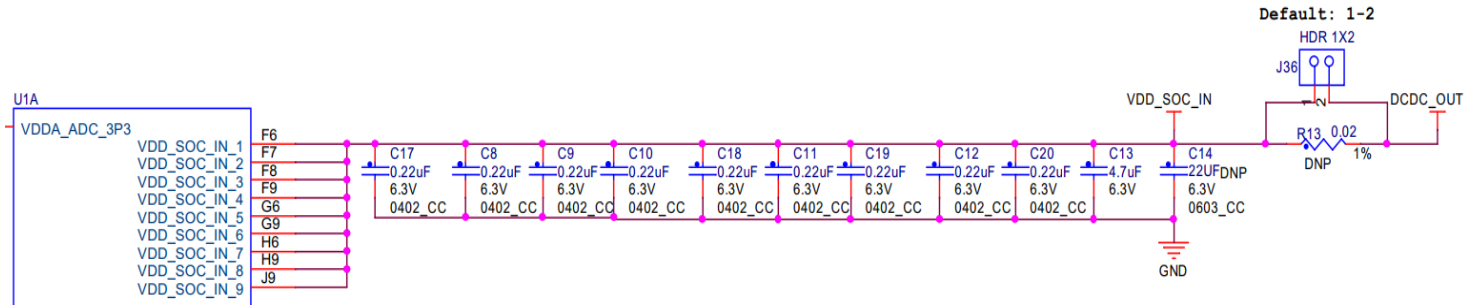


Table 10. Operating ranges

Parameter Description	Symbol	Operating Conditions	Min	Typ	Max[1]	Unit	Comment
Run Mode	VDD_SOC_IN	M7 core at 528 MHz	1.15	—	1.26	V	—
		M7 core at 132 MHz	1.15	—	1.26		
		M7 core at 24 MHz	0.925	—	1.26		
IDLE Mode	VDD_SOC_IN	M7 core operation at 528 MHz or below	1.15	—	1.26	V	—
SUSPEND (DSM) Mode	VDD_SOC_IN	—	0.925	—	1.26	V	Refer to Table 13 Low power mode current and power consumption
SNVS Mode	VDD_SOC_IN	—	0		1.26	V	—



Notes: DCDC_SENSE管脚因为是内部DCDC的电压反馈，所以其与DCDC_OUT的输出之间不要串联任何电阻，否则会导致输出电压不准

RT内部DCDC设计要点（二）------ PSWITCH RC延迟电路原理

RC电路参数的选择：

内部DCDC是认为pswitch到1.65V即可开启，Pswitch外部加的延时是延时0-1.65V的RC常数，为了确保在开启DCDC前DCDC输入已经到达2.8V-3.3V

所以需要PSwitch延时一定时间长度。

Vt=V0+（Vu-V0）* [1-exp(-t/RC)]

t = RC × Ln[(V1 - V0)/(V1 - Vt)]

V0=0V的话Vt=E*[1-exp(-t/RC)]

完全充满，Vt接近E，时间无穷大；

当t= RC时，电容电压=0.63E；

当t= 2RC时，电容电压=0.86E；

EVKB 上 pswitch上RC为30K 0.22uF

如果Vt=1.65V=E/2,需要t=-RC*ln(0.5)=RC*0.69=4.5ms, 这个就是EVB上的PSWITCH到达DCDC开启时间。实际上考虑板级电容，和上电时间，可能会稍微更长一点。那外部DCDC输入电压在这个时间内达到2.8V以上即可

Designer的最终要求：

When the DCDC_IN reach 3.3V, the PSWITCH voltage should be less than 0.3*DCDC_IN.

即：DCDC_IN满足条件的电压时，PSWITCH不高于1V， t = RC × Ln[(3.3)/(3.3 - 0.99)]=0.36RC

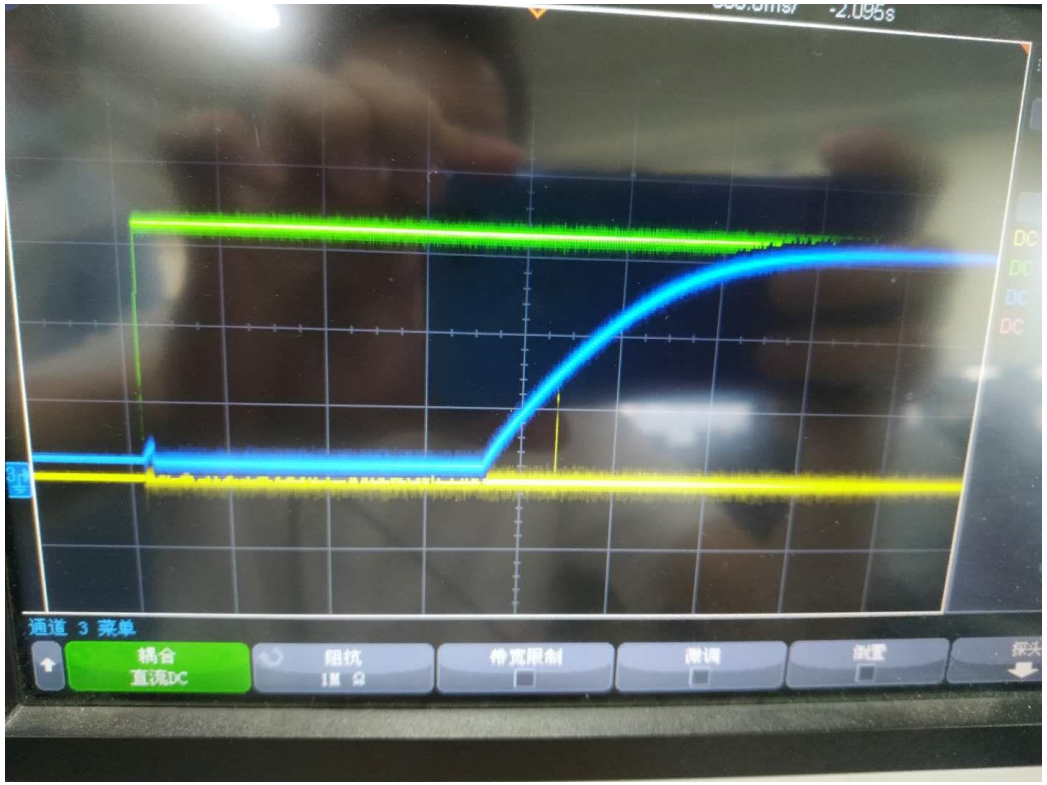
反推如果客户板子DCDC上电时间是50ms，需要调整PSWICTH外部的RC，满足

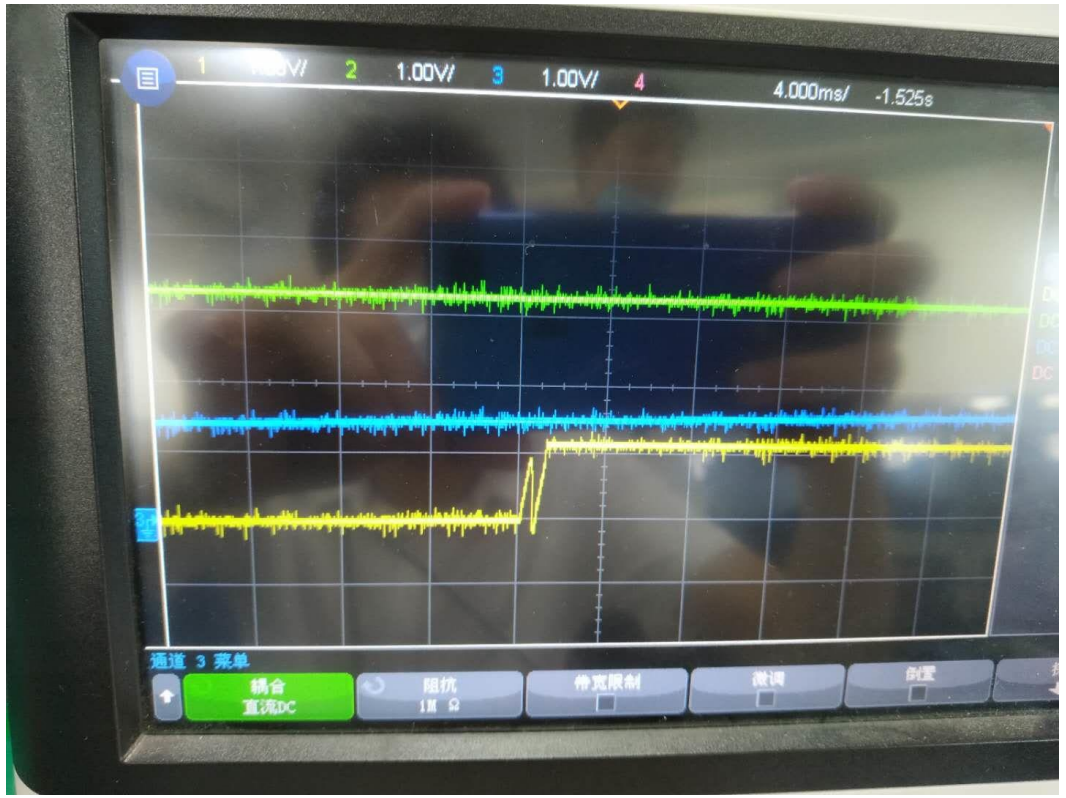
designer的要求0.36*RC：RC=139ms

RT内部DCDC设计要点（三）------ PSWITCH上升斜率不能太慢

PSWITCH的上升斜率也不能太慢，否则会导致DCDC在开启状态下对临界开关产生疑问。这样会尝试开启一次，有可能会再次开启也有可能就失败了。

对于PSWITCH外部有电压检测芯片检测的情况，PSWITCH的阻容不需要关心延时时间，这个上升斜率可以尽可能快。





绿色为3v3，蓝色为PSWITCH，黄色为1.2v，左右两图为同样一个很大的RC常数（几百ms级别），左图为启动失败，右图为启动成功

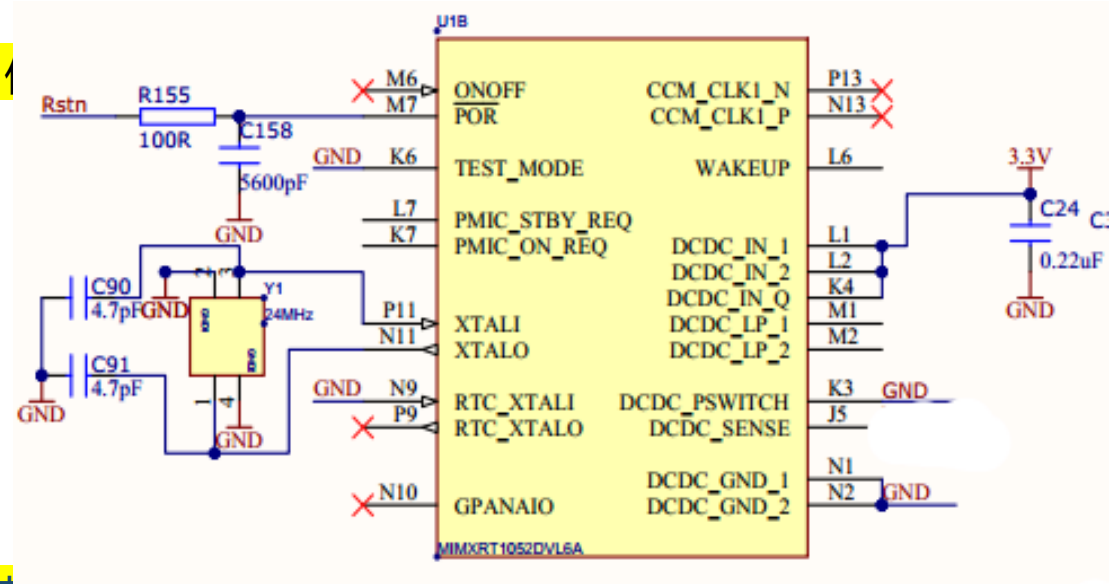
RT内部DCDC设计要点（四）------ 使用外部独立电源供电VDD_SOC_IN

- 不使用内部DCDC提供VDD_SOC_IN的电压，如果板上有匹配的电压，采用外部独立的电压源作为该电压输入也是可以的。优点在于省掉一个电感，减小板子空间。

Connect DCDC_IN as 3V3, DCDC_PSWITCH to GND, DCDC_LP float

如果不需要使用内部 DCDC 而是通过外部电源给 VDD_SOC_IN 上电的话，
DCDC_IN 供电并将 DCDC_PSWITCH 接地，同时保持 DCDC_LP 浮空；

同时注意将软件SDK中，关于DCDC初始化的相关代码屏蔽掉！



当然需要注意这里面潜在的一些缺陷：

1.该电压不能实时调整，对于低主频下的低功耗不利；

2.上电的时候确保3.3V先于1.2V上电。这样确保外围电路比如flash优先获得电源。

RT的复位电路设计

- RT的内部复位是由外部输入与内部逻辑相与作用的结果，内部的复位电路在检测到VDD_SOC_IN达到0.8V之后，延时1ms后会拉高释放复位逻辑，所以理论上可以不需要外部复位监控芯片，POR_B脚直接上拉4.7k到VDD即可。不过从可靠性设计上来考虑，一般建议外加电压监控复位芯片与POR_B管脚相连，原因主要有三点：

（1）RT的最小可工作核电压为0.925v，而内部复位逻辑待核电压升到0.8v后延迟1ms就会释放，如果VDD_SOC_IN上升缓慢在1ms时间内达不到0.925v的话则CPU有跑飞风险，因为复位逻辑释放后CPU已经可以跑了；

（2）当DCDC_IN掉到2.6v以下之后，DCDC功能会关掉，所以导致VDD_SOC_IN开始下降，而由于RT没有LVD功能（欠压检测复位）,出现此情况时内部复位逻辑没有assert复位信号，当核电压低于1.15v之后会可能导致CPU异常跑飞，所以外加电压监控复位芯片是必要的；

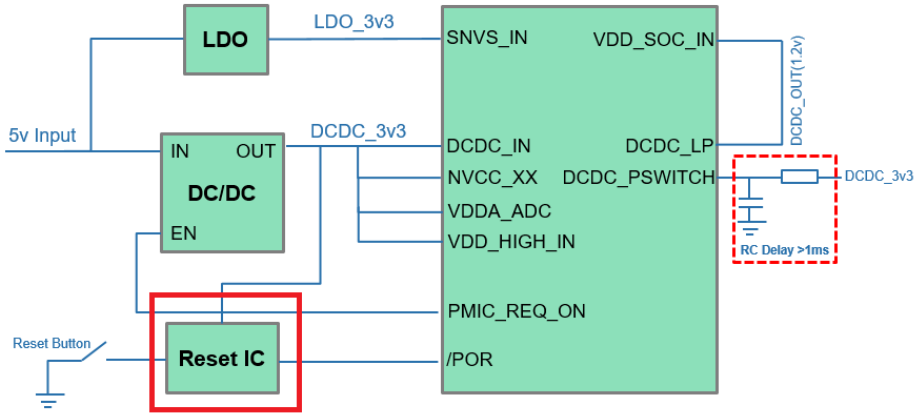
（3）Boot Config模式是在POR_B管脚的上升沿采样Boot_Cfg几个管脚决定的，复位芯片一般会在电压上升到正常值之后最大拉低POR_B140ms，可以保证采样boot config信号相比于不采用复位芯片的时序会更加稳定；

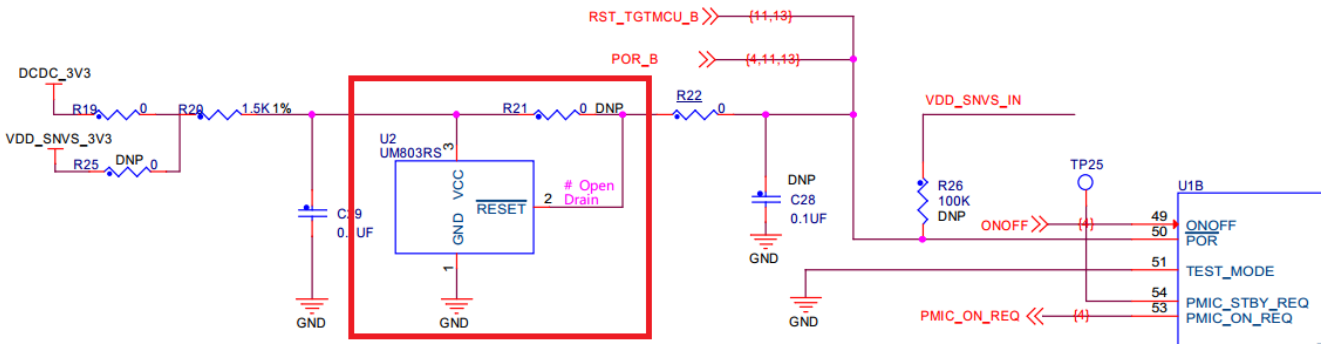
21.6 Power-On Reset and power sequencing

This module generates an internal POR_B signal that is logically AND'ed with any externally applied SRC_POR_B signal. The internal POR_B signal will be held low until all of the following conditions are met:

- 4ms after the external power supply VDDHIGH_IN is valid
- 1ms after the VDD_SOC_IN supply is valid

The 4ms and 1ms delays are derived from counting the 32 kHz RTC clock cycles; the accuracy depends on the accuracy of the RTC. When the RTC crystal is either absent or in the process of powering up, an internal ring oscillator will be the source of RTC, which is not as accurate as the crystal.





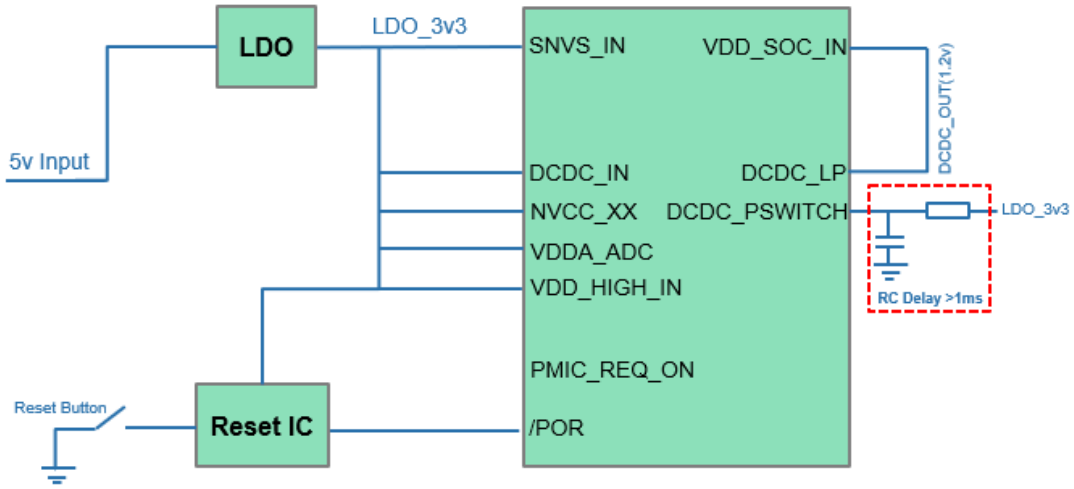
RT的复位电路与PSWITCH信号结合

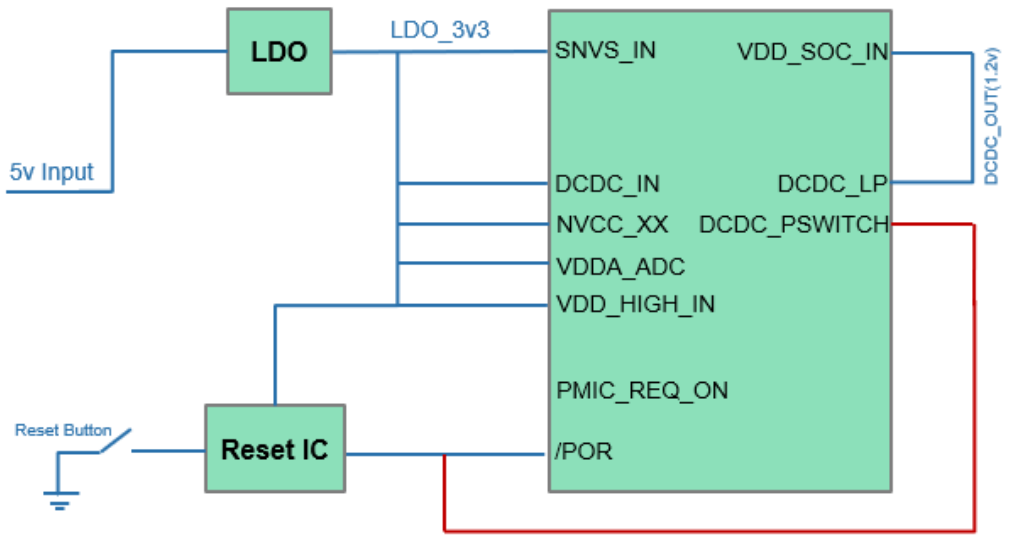
- 由于PSWITCH的特殊设计（必须延迟1ms以上），它带来潜在的两个限制：

（1）当IO上先有电而RT的电源后上电，该情况会导致内部DCDC输出1.2v核电压失败，原因是当IO上有电压之后会漏到DCDC_IN电源上，而PSWITCH由于与DCDC_IN电源相连同样会有残压存在（这个漏电压跟外部IO驱动能力有关系，一般1v~2v之间），此时DCDC_IN再上电由于PSWITCH的RC延迟作用失效而导致DCDC工作失败；

（2）非常快速开关RT的电源或者电源有快速跌落的情况，当关掉3v3或者电源跌落到2.6v以下时，1.2v内核电源关闭，DCDC_IN开始下降，PSWITCH由于RC延迟电路的缘故导致其下降速度慢于DCDC_IN的速度，在PSWITCH信号低于1v之前外部电源重新上电，此种情况也会导致RC延迟失效，进而DCDC启动失败；

如果对以上两点有特别要求的用户可以使用下面右图的连接方法，即PSWITCH与POR_B短接，当DCDC_IN的电源关闭或者跌落时，复位芯片会产生拉低，把PSWITCH和POR_B迅速放电到低电平，待DCDC_IN回升至正常电压后，复位芯片会延迟140ms后拉高，同样可以满足DCDC上电时序。

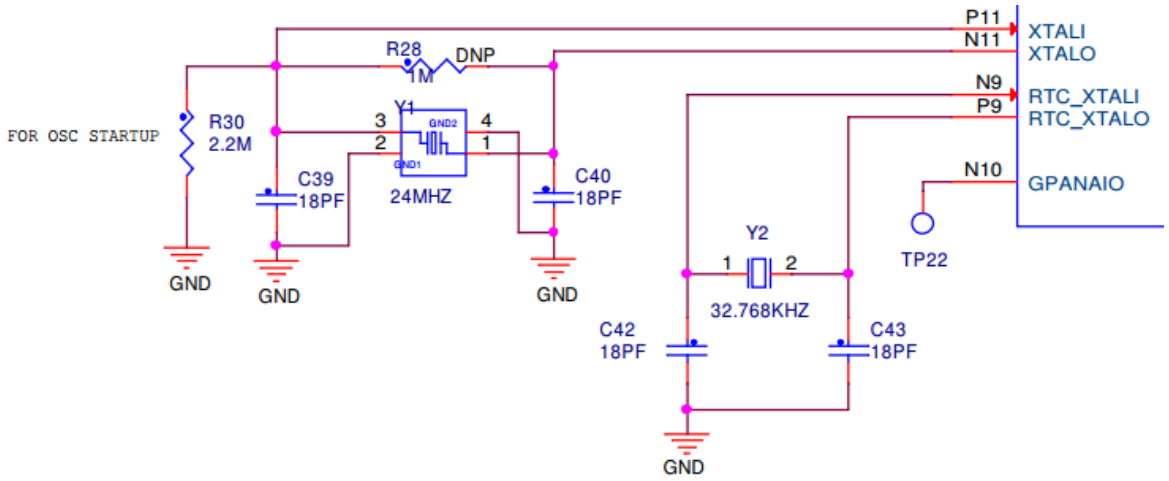


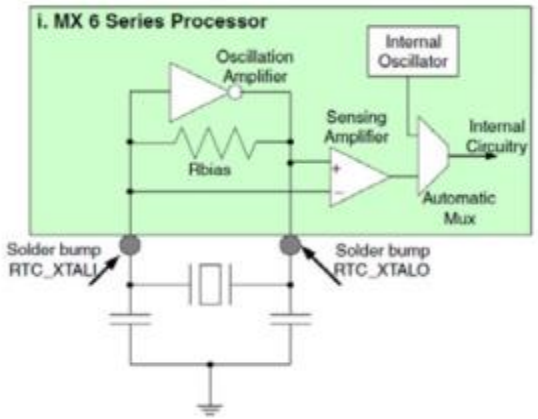


RT的时钟电路设计（一）

RT的外部时钟输入有两个，其一是32.768KHz给SNVS控制域和RTC模块，另一个则是24MHz时钟作为同步时钟给ROM和内部PLL使用。RT内部时钟仅在系统刚启动时使用，当检测到外部有时钟时会自动切换到外部时钟源，如果不需要低功耗RTC应用的情况下，RTC时钟可以使用内部，此时RTC_XTALI需要接地，RTC_XTALO浮空，而24MHz时钟则建议一定要使用外部高精度晶体。

Notes: 由于内部看门狗的时钟源为RTC_CLK，所以如果选择不使用外部32k晶体的话，内部看门狗的超时时间建议设置长一些，避免由于32k时钟内部精度差而导致软件喂狗慢而误触发看门狗超时复位。



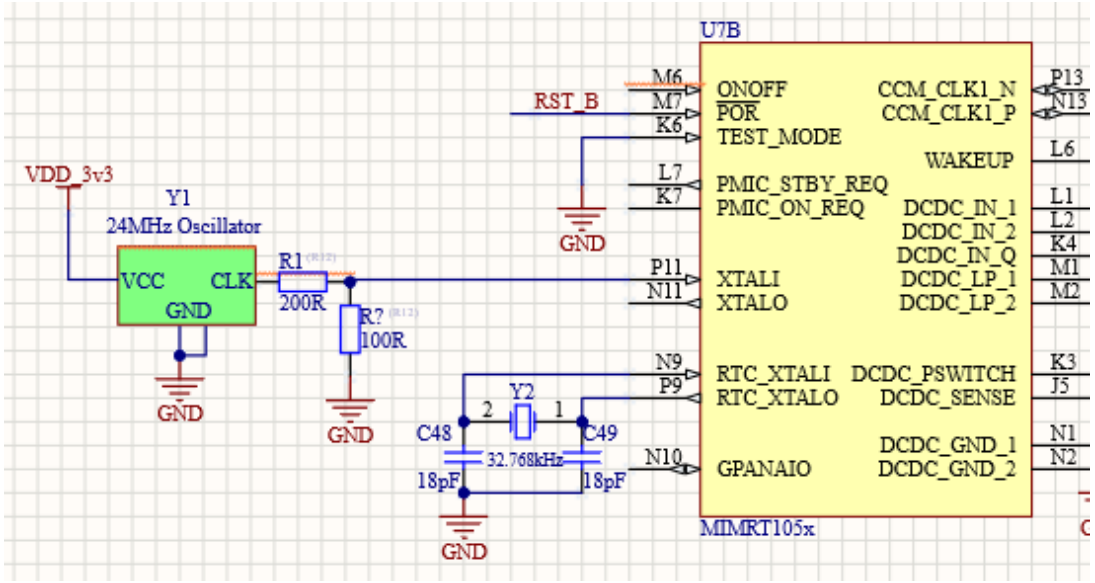
信号名称	推荐连接	说明
1.RTC_XTALI/RTC_XTALO	 对于高精度 32.768-kHz 振荡器，在 RTC_XTALI 和 RTC_XTALO 之间连接一个晶振。选择最大 ESR（等效串联电阻）为 100 k 的晶振，并遵循制造商关于负载电容的建议。不要使用外部偏置电阻，因为芯片已经内置了该偏置电阻。	要达到确切的振荡频率，必须减少电路板电容，以解决电路板和芯片的寄生效应。集成振荡放大器具有自偏置性，但相对较弱。必须注意将来自 RTC_XTALI 和 RTC_XTALO 的寄生泄漏限制为电源或接地（大于 100 M）。这会使放大器除偏并降低启动裕量。
	对于外部 kHz 时钟源（如果将外部时钟源直接输入到芯片时钟管脚），RTC_XTALI 可以通过直流耦合驱动并使 RTC_XTALO 浮空，也可以由互补信号驱动。	如果要将外部低频时钟馈入 RTC_XTALI，RTC_XTALO 引脚必须保持未连接状态或由互补信号驱动。此强制时钟的逻辑电平不得超过 VDD_SNVS_CAP 电平，在典型条件下频率应小于 100 kHz。
	芯片内置大约 40KHz 片上振荡器。如果 RTC_XTALI 连接到 GND 且 RTC_XTALO 浮空，则片上振荡器自动启动。	当不需要高精度实时时钟时，系统可以使用片上 40kHz 振荡器。容差为±50%。环形振荡器的启动速度比外部晶振快，并在外部晶振达到稳定振荡之前使用。如果在 RTC_XTALI 未检测到时钟，环形振荡器也会自动启动。

RT的时钟电路设计（二）

- 对于电磁干扰敏感的场合，使用外部有源晶振能够获得更好的EMC性能，不过对有源晶振来说，要求XTALI的输入电压必须符合(0.8*NVCC_PLL ~ NVCC_PLL)的电压范围，请参阅数据手册。

- 虽然手册见右图中提到有源晶振的接法可以有三种，但是一般建议第二种（注意与Datasheet的不同，以此文为准），即只使用从XTALI输入符合电平要求的有源时钟即可，XTALO保持悬空。

Notes: 由于1.1v的有源晶振很难买到，可以使用下图设计方式选择100欧和200欧电阻分压的方式。



15.4.2 Bypass Configuration (24 MHz)

If it is desired to drive the chip with an external clock source, then the 24 MHz oscillator could be driven in one of three configurations using a nominal 1.1V source.

1. A single ended external clock source can be used to overdrive the output of the amplifier (XTALO). Since the oscillation sensing amplifier is differential, the XTALI pin should be externally floating and capacitively loaded. The combination of the internal biasing resistor and the external capacitor will filter the signal applied to the XTALO pin and develop a rough reference for the sensing amplifier to compare to.
2. A single ended external clock source can be used to drive XTALI. In this configuration, XTALO should be left externally floating.
3. A differential external clock source can be used to drive both XTALI and XTALO.

Generally, configuration 2 is anticipated to be the most used configuration, but all three configurations may be utilized.

4.3.1.1 XTALI and RTC_XTALI (clock inputs) DC parameters

Table 21 shows the DC parameters for the clock inputs.

Table 21. XTALI and RTC_XTALI DC parameters[1]

Parameter	Symbol	Test Conditions	Min	Max	Unit
XTALI high-level DC input voltage	Vih	—	0.8 x NVCC_PLL	NVCC_PLL	V
XTALI low-level DC input voltage	Vil	—	0	0.2	V
RTC_XTALI high-level DC input voltage	Vih	—	0.8	1.1	V
RTC_XTALI low-level DC input voltage	Vil	—	0	0.2	V

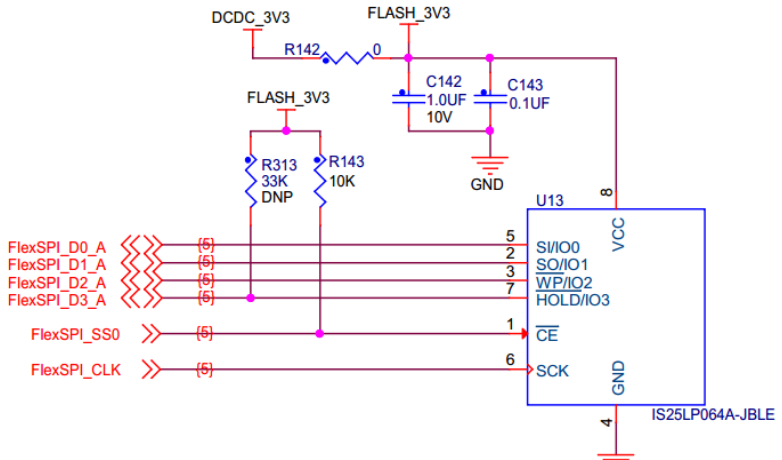
[1] The DC parameters are for external clock input only.

SPI Flash接口设计

- For QSPI Flash as XIP function, to archieve highest performance it is recommended to left FLEXSPI_A_DQS（GPIO_SD_B1_05）pin floating and config sample clock source as 01-loopback from DQS pad mode to archieve max 133MHz Flexspi_clk. Besides, it is also recommend to pull-up QSPI flash pin1(CE) and pin7(hold) to make sure QSPI Flash boot up stability. (Note: SEMC_DQS is also recommend to be floating to archieve max speed)
- 注意:HOLD和WP的上拉电阻是需要的。并不是所有SPI Flash都有内部上拉;
- 如果EMC测试时，对外电磁辐射频率点与Flexspi的时钟接近（或倍数），可以软件上降低对应的时钟flexspi_clk的输出Drive Strength Field

(其他高速接口设置类似)

FLEXSPI A	FLEXSPI_A_DATA0	GPIO_AD_B1_02	ALT1
		GPIO_SD_B1_08	ALT1
	FLEXSPI_A_DATA1	GPIO_AD_B1_04	ALT1
		GPIO_SD_B1_10	ALT1
	FLEXSPI_A_DATA2	GPIO_AD_B1_03	ALT1
		GPIO_SD_B1_09	ALT1
	FLEXSPI_A_DATA3	GPIO_AD_B1_00	ALT1
		GPIO_SD_B1_06	ALT1
	FLEXSPI_A_DQS	GPIO_SD_B1_05	ALT1
	FLEXSPI_A_SCLK	GPIO_AD_B1_01	ALT1
		GPIO_SD_B1_07	ALT1
	FLEXSPI_A_SS0_B	GPIO_AD_B1_05	ALT1
		GPIO_SD_B1_11	ALT1



5-4 RXCLKSRC	Sample Clock source selection for Flash Reading Refer RX Clock Source Features for more details 00b - Dummy Read strobe generated by FlexSPI Controller and loopback internally. 01b - Dummy Read strobe generated by FlexSPI Controller and loopback from DQS pad. 10b - Reserved 11b - Flash provided Read strobe and input from DQS pad

4.5.2.1.1 SDR mode with FlexSPI*n*_MCR0[RXCLKSRC] = 0x0, 0x1

Table 35. FlexSPI input timing in SDR mode where FlexSPI*n*_MCR0[RXCLKSRC] = 0X0

Symbol	Parameter	Min	Max	Unit
—	Frequency of operation	—	60	MHz
T_{IS}	Setup time for incoming data	8.67	—	ns
T_{IH}	Hold time for incoming data	0	—	ns

Table 36. FlexSPI input timing in SDR mode where FlexSPI*n*_MCR0[RXCLKSRC] = 0X1

Symbol	Parameter	Min	Max	Unit
—	Frequency of operation	—	133	MHz
T_{IS}	Setup time for incoming data	2	—	ns
T_{IH}	Hold time for incoming data	1	—	ns

Boot mode Config Design

- 在RT EVK的原理图设计上都能看到如下启动选项的设计：

  (1)首先根据BOOT_MODE的配置，在上电时确定以何种方式启动：

Table 9-2. Boot MODE pin settings

BOOT_MODE[1:0]	Boot Type
00	Boot From Fuses
01	Serial Downloader
10	Internal Boot
11	Reserved

第一版原理图设计的时候可以使用跳帽或者切换开关，使用三极管电路设计，使BOOT mode在01和10键值之间，直接一键简便切换。

(2) BOOT_CFG[0-11]确定当选择Internal boot的情况下，从外部何种设备启动。外部管脚与Fuse map一一对应，无论是Boot from fuse和Internal Boot都具体可以查阅下表的Fuse Map信息。默认的全0状态，代表从外部的地址位为24bit的QSPI启动。

(3) BOOT_MODE和BOOT_CFG都是内部100K电阻下拉的，所以EVK上的下拉电阻不是必须的。浮空状态在启动时电平即为下拉，启动后恢复为正常状态。如果需要复用这些管脚，可以将其使用为输出或者输入带隔离等应用，确保启动时，不为外部设备所影响。

(4) BOOT_MODE和BOOT_CFG管脚状态均是在POR_B管脚的上升沿时由内部逻辑锁存住电平状态，之后这些管脚可以正常做GPIO功能使用，但是建议BOOT_MODE管脚尽量不要另做它用。

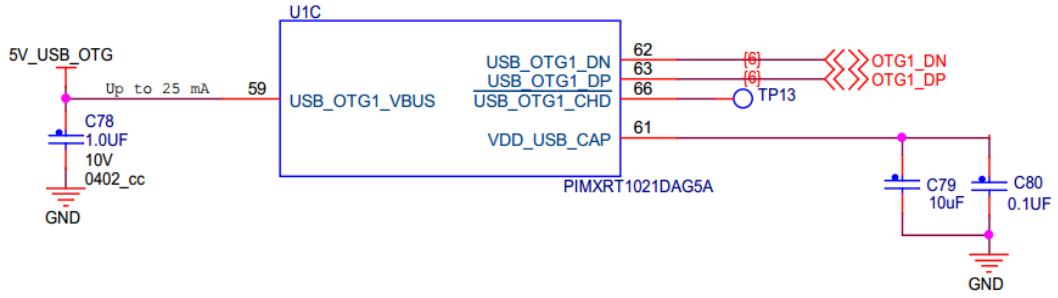
Notes: POR_B上升沿时刻锁存的电平状态可以通过SRC_SBMR1和SRC_SBMR2这两个寄存器读取，用来检查刚启动时Boot相关管脚的电平，在Boot启动失败时可以提供线索，即在Jlink Command命令窗口里先connect上RT的内核，然后通过mem32命令读取这两个寄存器的内容来判断启动失败的可能原因。

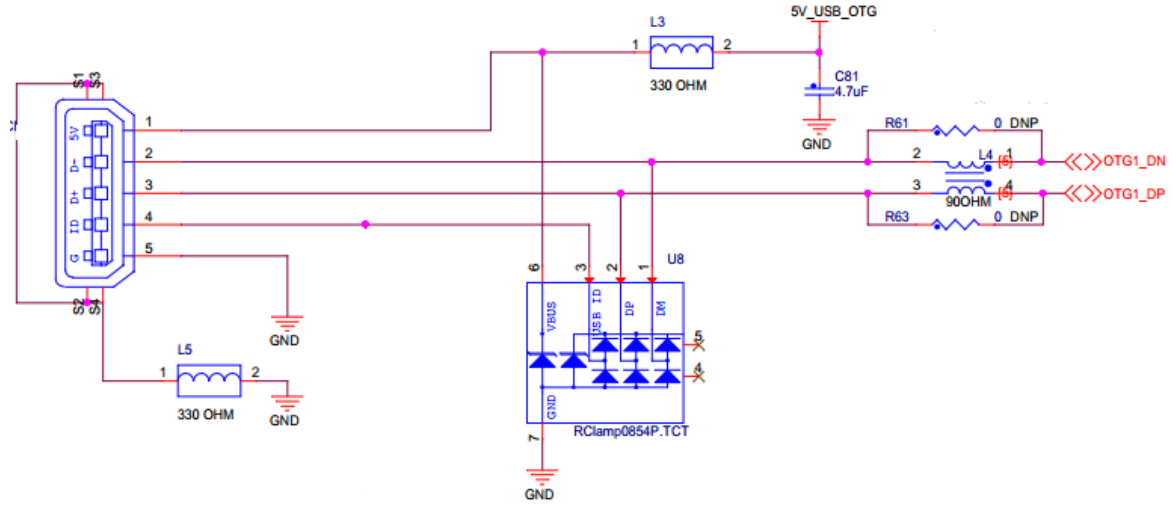
Notes: BT_CFG are assigned to different GPIOs on RT1050 (GPIO_B0_04:15) and 1020(GPIO_EMC18:27)

电路设计时，非必要建议采用外部管脚启动，这样配置灵活且不需要烧写fuse☺

USB Design

- USB_OTG1 interface or LPUART1(GPIO_AD_B0_12 and GPIO_AD_B0_13) should be reserved for NXP flashloader tools, including firmware update, security and Fuse by setting BOOT MODE as serial download mode. And of course USB_OTG1 can be used as normal USB 2.0 function. Besides, one thing should be taken care that USB TVS protection diode should select corresponding part for full-speed and high-speed USB application.;
- For app without USB function, the USB pins can be left floating;
- For USB circuit hardware design guide, please refer to RT105060 hardware user guide.





8.1 Chip-specific Boot Information

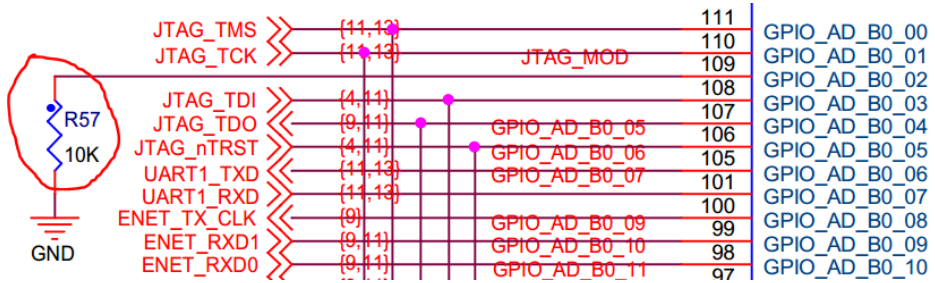
This device has various peripherals supported by the ROM bootloader.

Table 8-1. ROM Bootloader Peripheral PinMux

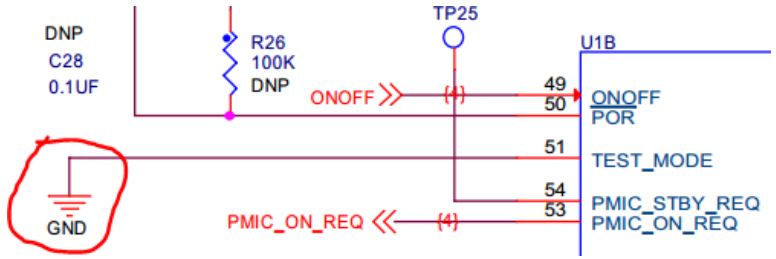
Peripheral	Instance	Port (IO function)	PAD	Mode
LPUART	1	LPUART1_TX	GPIO_AD_B0_12	ALT2
		LPUART1_RX	GPIO_AD_B0_13	ALT2

Small Tips Design

1. JTAG_MOD pin must be tied to GND with a pull-down resistor to make sure JTAG/SWD debug working normally;



2. TEST_MODE pin is reserved for NXP factory manufacturing use, suggest to tie to GND directly;



3. Both Flexspi_DQS and SEMC_DQS pad are recommended to be floating and do not used for other function if external memory has not this DQS dedicated pin;

Small Tips Design

4. JTAG/SWD电路设计：一般建议JTAG或者SWD管脚保持悬空即可，因为其内部已经默认有上下拉电阻使能了，如果一定要外加上下拉也一定要注意跟内部上下拉保持一致，避免管脚由于错误的上下拉导致中间电平的出现。

Notes: RT10xx系列里，除了RT1050的CLK管脚是内部上拉之外，其他系列的CLK管脚均为内部下拉100k电阻

RT1050 JTAG/SWD

Table 4. JTAG Controller interface summary

JTAG	I/O Type	On-chip Termination
JTAG_TCK SWD_CLK	Input	47 kΩ pull-up
JTAG_TMS SWD_DIO	Input	47 kΩ pull-up
JTAG_TDI	Input	47 kΩ pull-up
JTAG_TDO	3-state output	Keeper
JTAG_TRSTB	Input	47 kΩ pull-up
JTAG_MOD	Input	100 kΩ pull-up

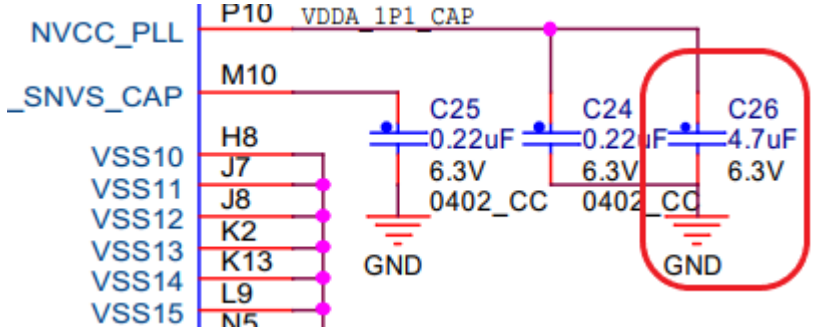
RT1020/RT1060 JTAG/SWD

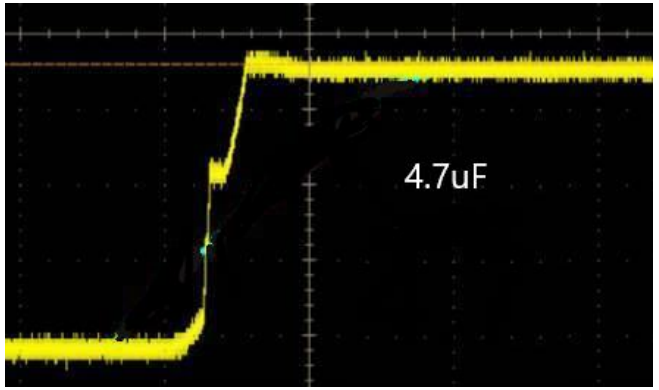
Table 5. JTAG controller interface summary

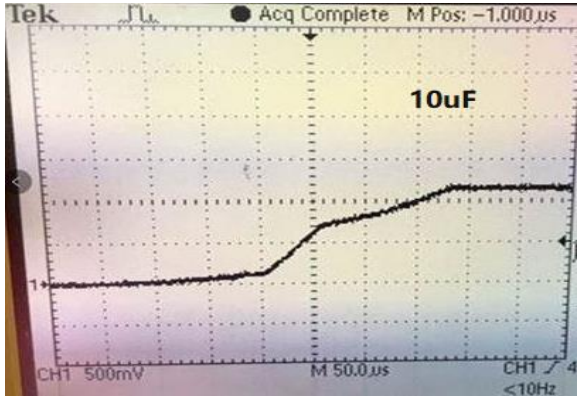
JTAG	I/O type	On-chip termination
JTAG_TCK SWD_CLK	Input	100 kΩ pull-down
JTAG_TMS SWD_DIO	Input	47 kΩ pull-up
JTAG_TDI	Input	47 kΩ pull-up
JTAG_TDO	3-state output	Keeper
JTAG_TRSTB	Input	47 kΩ pull-up
JTAG_MOD	Input	100 kΩ pull-down

Small Tips Design

5. NVCC_PLL的稳压电容建议改成10uF，现有的EVK设计4.7uF会存在PLL上升电压不稳定的情况，改成10uF可以改善很多;







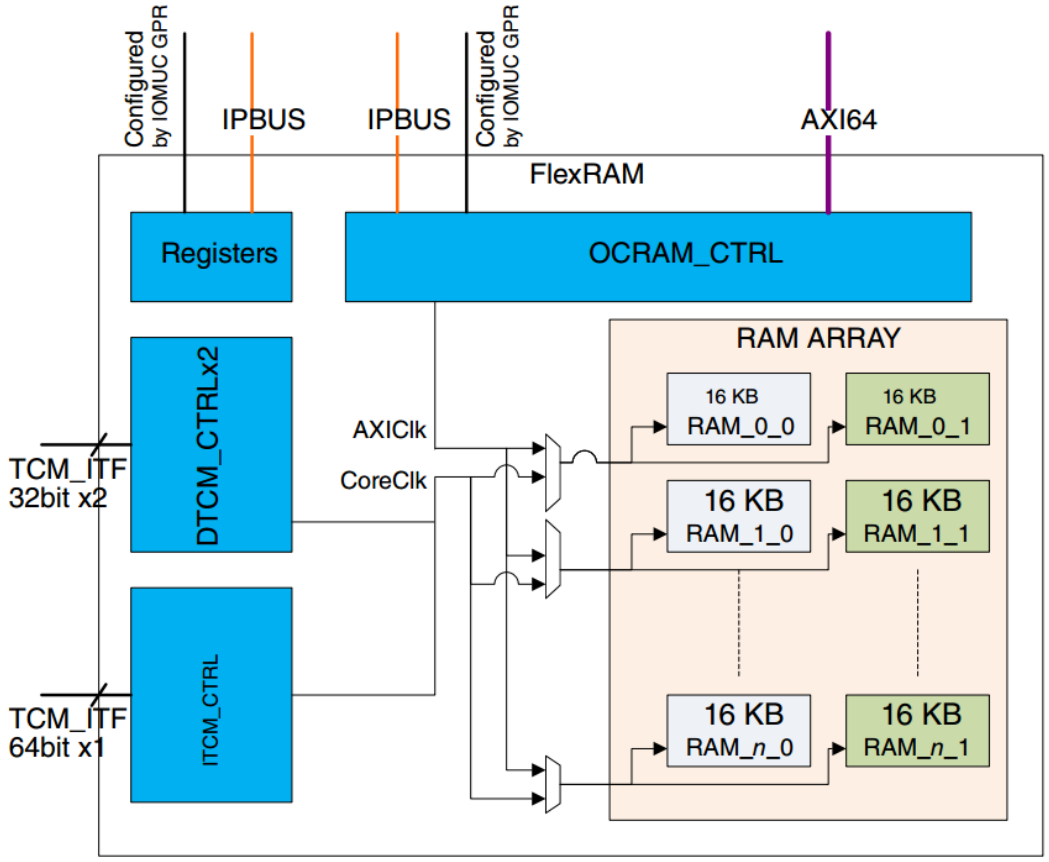
IMX RT Design Tips —Software

动态重分配FlexRAM(ITCM/DTCM/OCRAM)

- i.MXRT1050 has total 512KB FlexRAM, default 128KB ITCM, 128KB DTCM and 256KB OCRAM (half on RT1020 and extra 512KB OCRAM on RT1060). While in many cases the default distribution can't meet customer requirement. Thanks to RT FlexRAM's Flex, it can be partitioned with different size with flexibility by fixed step size(32KB for RT1050, 16KB for RT1020).

- Two options to redistribute flexRAM, one is by setting eFuse(OTP, config only once), another is by setting IOMUXC_GPR register dynamically which is more flexible. Here I take the latter one as example at Keil.

FlexRAM	Memory Map
ITCM	0x0000_0000~
DTCM	0x2000_0000~
OCRAM	0x2020_0000~



动态重分配FlexRAM(ITCM/DTCM/OCRAM)

1. Firstly, the ITCM, DTCM and OCRAM's memory map should be well noted as below;

FlexRAM	Memory Map
ITCM	0x0000_0000~
DTCM	0x2000_0000~
OCRAM	0x2020_0000~

2. IOMUXC_GPR_GPR14, 16 and 17, these three registers need to be learned.

(a) CFGITCMSZ and CFGDTCMSZ in GPR14 means max configurable size, so it is ok to just set them as 0b1010(512KB);

(b) BANK_CFG_SEL in GPR16 controls FlexRAM bank config source, default use fuse value to config(the default fuse configs 128KB ITCM, 128KB DTCM and 256KB OCRAM), we should change this BANK_CFG_SEL bit to use FLEXRAM_BANK_CFG which is in GPR17 register to config FlexRAM partition, so GPR16 should be ORed with 0x0000_0007;

IOMUXC_GPR_GPR14 field descriptions (continued)

Field	Description
	0110 32 KB 0111 64 KB 1000 128 KB 1001 256 KB 1010 512 KB other reserved
19–16 CM7_ CFGITCMSZ	ITCM total size configuration 0000 0 KB (No ITCM) 0011 4 KB 0100 8 KB 0101 16 KB 0110 32 KB 0111 64 KB 1000 128 KB 1001 256 KB 1010 512 KB other reserved

IOMUXC_GPR_GPR16 field descriptions

Field	Description
31–7 CM7_INIT_VTOR	Vector table offset register out of reset. See the ARM v7-M Architecture Reference Manual for more information about the vector table offset register (VTOR).
6–3 -	This field is reserved. Reserved
2 FLEXRAM_ BANK_CFG_SEL	FlexRAM bank config source select 0 use fuse value to config 1 use FLEXRAM_BANK_CFG to config
1 INIT_DTCM_EN	DTCM enable initialization out of reset. **NOTE:** When a TCM is enabled, the corresponding CFG*TCMSZ register must NOT be set to 0'b0000 0 DTCM is disabled 1 DTCM is enabled
0 INIT_ITCM_EN	ITCM enable initialization out of reset. **NOTE:** When a TCM is enabled, the corresponding CFG*TCMSZ register must NOT be set to 0'b0000 0 ITCM is disabled 1 ITCM is enabled

动态重分配FlexRAM(ITCM/DTCM/OCRAM)

(c) FLEXRAM_BANK_CFG control which RAM bank using for ITCM/DTCM/OCRAM. The below table in AN12077 can be taken as reference for 16 possible configuration value, but it is not limited with these 16 kinds;

34.4.18 GPR17 General Purpose Register (IOMUXC_GPR_GPR17)

GPR Register

Address: 400A_C000h base + 44h offset = 400A_C044h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	FLEXRAM_BANK_CFG																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC_GPR_GPR17 field descriptions

Field	Description
FLEXRAM_BANK_CFG	FlexRAM bank config value GPR_FLEXRAM_BANK_CFG[2*n*+1 : 2*n*], where n = 0, 1, ..., 15 • 00: RAM bank *n* is not used • 01: RAM bank *n* is OCRAM • 10: RAM bank *n* is DTCM • 11: RAM bank *n* is ITCM

Table 1. 16 possible configurations of FlexRAM banks using fuse value on i.MX RT1050

	FUSE FlexRAM Configuration Value	IOMUXC_GPR_GPR17 (FLEXRAM_BANK_CFG) (binary)	Bank															OCRAM [kB]	DTCM [kB]	ITCM [kB]
	0x6D0 [19:16]		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15			
0	0b0000	01010101101011111111101001010101	O	O	O	O	D	D	I	I	I	I	D	D	O	O	O	O	256	128	128
1	0b0001	01010101010110101111101001010101	O	O	O	O	D	D	I	I	I	D	D	O	O	O	O	O	320	128	64
2	0b0010	01011010111111111111111101001010	O	O	D	D	I	I	I	I	I	I	I	I	D	D	O	O	128	128	256
3	0b0011	01010101010101011101010010101	O	O	O	D	D	D	D	I	O	O	O	O	O	O	O	O	352	128	32
4	0b0100	01010101010111111111101001010101	O	O	O	O	D	D	I	I	I	I	O	O	O	O	O	O	320	64	128
5	0b0101	01010101010101011111101001010101	O	O	O	O	D	D	I	I	I	O	O	O	O	O	O	O	384	64	64
6	0b0110	01010101111111111111111101001010	O	O	D	D	I	I	I	I	I	I	I	I	O	O	O	O	192	64	256
7	0b0111	11111111111111111111111111101010	O	O	I	I	I	I	I	I	I	I	I	I	I	I	I	I	64	0	448
8	0b1000	01011010101011111111101010100101	O	O	D	D	D	D	I	I	I	I	D	D	D	D	O	O	128	256	128
9	0b1001	01010101101010101111101010100101	O	O	D	D	D	D	I	I	I	D	D	D	D	O	O	O	192	256	64
10	0b1010	10101010111111111111111110100101	O	O	D	D	I	I	I	I	I	I	I	I	D	D	D	D	64	192	256
11	0b1011	10101010101010101010101010010101	O	O	D	D	D	D	D	D	D	D	D	D	D	D	D	D	64	448	0
12	0b1100	01010101010101011111111101010101	O	O	O	O	I	I	I	I	O	O	O	O	O	O	O	O	384	0	128
13	0b1101	01010101010101010101011110010101	O	O	O	D	I	I	O	O	O	O	O	O	O	O	O	O	448	32	32
14	0b1110	01010101010111111111111111110101	O	O	I	I	I	I	I	I	I	I	O	O	O	O	O	O	256	0	256
15	0b1111	01010101010101010101010101010101	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	512	0	0
			O - OCRAM, D - DTCM, I - ITCM																		

3. In keil, two must actions need to modify accordingly, the scatter file and the above three registers' config. Scatter file should match the new RAM partition and three GPR register should config before __main in keil, so it is ok to add them at Systeminit() while it is strongly to modify the flexram config at Reset_handler entry in startup_MIMXRT10xx.s in AN12077 AN document;

gpio_led_output.c | MIMXRT1052xxxxx_ram.scf | evkbimxrt1050_ram.ini | system_MIMXR

```
56
57  #define m_interrupts_start              0x00000000
58  #define m_interrupts_size               0x00000400
59
60  #define m_text_start                    0x00000400
61  #define m_text_size                     0x0003FC00 /*256KB ITCM*/
62
63  #define m_data_start                    0x20000000
64  #define m_data_size                     0x00030000 /*192KB DTCM*/
65
66  #define m_data2_start                   0x20200000
67  #define m_data2_size                    0x00010000 /*64KB OCRAM*/
```

gpio_led_output.c | MIMXRT1052xxxxx_ram.scf | evkbimxrt1050_ram.ini | system_MIMXRT1052.c

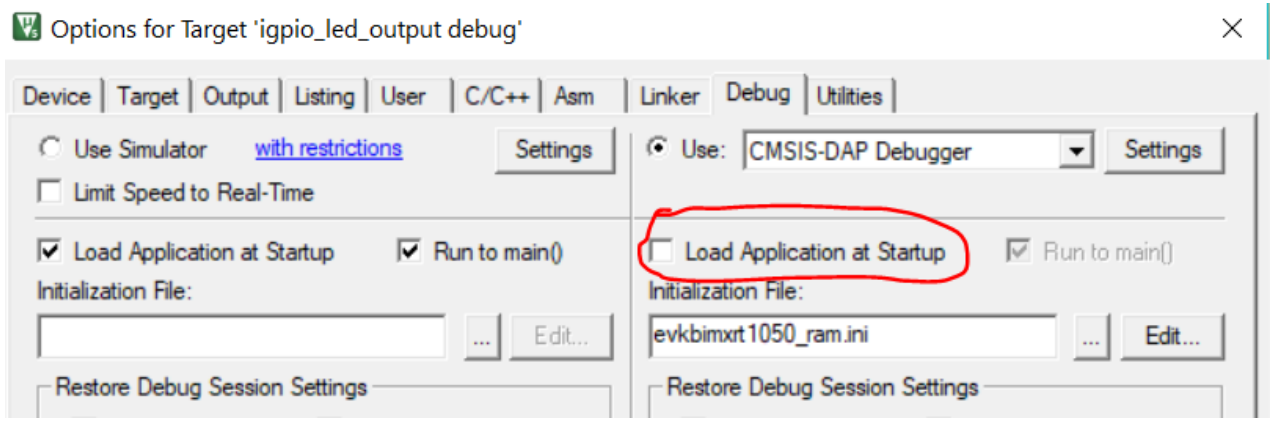
```
291 #if defined(__DCACHE_PRESENT) && __DCACHE_PRESENT
292     if (SCB_CCR_DC_Msk != (SCB_CCR_DC_Msk & SCB->CCR)) {
293         SCB_EnableDCache();
294     }
295 #endif
296   //SDRAM_Init();
297   IOMUXC_GPR->GPR14 |= 0x00AA0000;
298   IOMUXC_GPR->GPR16 |= 0x00000007;
299   IOMUXC_GPR->GPR17 = 0xAAFFFFA5;//256KB ITCM, 192KB DTCM, 64KB OCRAM
300
301   SystemInitHook();
302 }
```

动态重分配FlexRAM(ITCM/DTCM/OCRAM)

4. If we want to debug the code on new ITCM, one more action should be done. The Initialization file should also be change to init the GPR registers to make the new partition take effective before downloading the firmare. Besides, the below “load application at Startup” selection must be removed as it will ignore the init file to download the firmware;

gpio_led_output.c | MIMXRT1052xxxxx_ram.scf | evkbimxrt1050_ram.ini | system_MIMXRT1052.

```
68
69      if (dcdc_trim_loaded)
70      {
71          // delay about 400us till dcdc is stable.
72          _Sleep_(1);
73      }
74  }
75
76  FUNC void Setup (void) {
77    _loadDcdcTrim();
78    SP = _RDWORD(0x00000000);           // Setup Stack Pointer
79    PC = _RDWORD(0x00000004);           // Setup Program Counter
80     WDWORD(0xE000ED08, 0x00000000);    // Setup Vector Table Offset Regi
81    /* Config the new partition before downloding */
82    _WDWORD(0x400AC038, 0x00AA0000);//GPR14
83    _WDWORD(0x400AC040, 0x00000007);//GPR16
84    _WDWORD(0x400AC044, 0xAAFFFFA5);//GPR14
85  }
86
87  FUNC void OnResetExec (void)  {       // executes upon software RESET
88    Setup();                            // Setup for Running
89  }
90
91  Setup();                              // Setup for Running
92
93  LOAD %L INCREMENTAL                   // Download
```



5. Enjoy;

如何把函数放到RAM里运行

- 在很多应用场合里，都有这样的需求，即把部分函数放到RAM里运行，一是为了提升代码执行效率，另一个是某些功能限制（比如擦写flash的代码不能在flash里执行而只能拷贝到ram里），下面两图分配给出了Keil下和IAR下修改linker file方式把某个.c文件里的所有函数都拷贝到ram里执行。

MIMXRT1062xxxxx_flexspi_nor.scf | MIMXRT1062xxxxx_flexspi_nor.icf

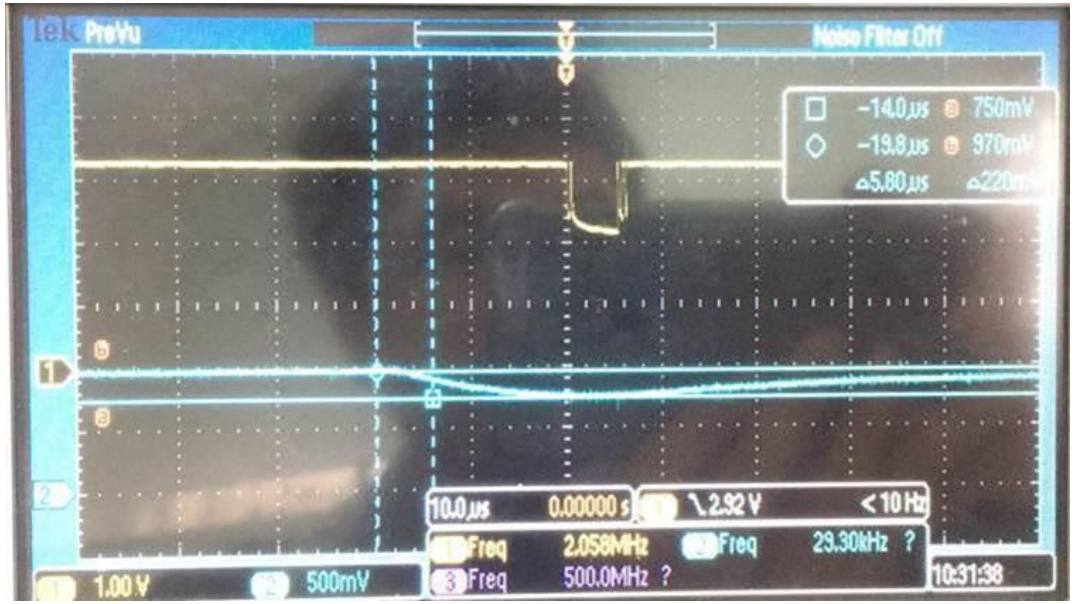
```
LR_m_text m_interrupts_start m_text_start+m_text_size-m_interrupts_start {   ; load region size_region
#endif
  VECTOR_ROM m_interrupts_start FIXED m_interrupts_size { ; load address = execution address
    * (.isr_vector,+FIRST)
  }
  ER_m_text m_text_start FIXED m_text_size { ; load address = execution address
    * (InRoot$$Sections)
    .ANY (+RO)
  }
  RW_m_data m_data_start m_data_size-Stack_Size-Heap_Size { ; RW data
    .ANY (+RW +ZI)
    /* Necessary to run flash routines from SRAM */
    flexspi_nor_flash_ops.o (+RO +RW +ZI)
    fsl_flexspi.o (+RO +RW +ZI)
    * (NonCacheable.init)
    * (*NonCacheable)
  }
  ARM_LIB_HEAP +0 EMPTY Heap_Size {    ; Heap region growing up
  }
  ARM_LIB_STACK m_data_start+m_data_size EMPTY -Stack_Size { ; Stack region growing down
  }
  RW_m_ncache m_data2_start EMPTY 0 {
  }
  RW_m_ncache_unused +0 EMPTY m_data2_size-ImageLength(RW_m_ncache) { ; Empty region added for MPU configuration
  }
}
```

MIMXRT1062xxxxx_flexspi_nor.scf | MIMXRT1062xxxxx_flexspi_nor.icf

```

initialize by copy {
  readwrite,
  /* Place in RAM flash and performance dependent functions */
  object flexspi_nor_flash_ops.o,
  object fsl_flexspi.o,
  section .textrw
};

do not initialize  { section .noinit };

place at address mem: m_interrupts_start    { readonly section .intvec };

place at address mem:m_boot_hdr_conf_start { section .boot_hdr.conf };
place at address mem:m_boot_hdr_ivt_start { section .boot_hdr.ivt };
place at address mem:m_boot_hdr_boot_data_start { readonly section .boot_hdr.boot_data };
place at address mem:m_boot_hdr_dcd_data_start { readonly section .boot_hdr.dcd_data };
```

RT低功耗的DCDC模式切换问题

- 默认的SDK低功耗模式切换存在潜在隐患，主要跟DCDC的DCM（非连续控制）和CCM（连续控制）模式有关系，CCM。DCDC的DCM模式，通常用在轻载情况下，优势为省电，但是负载响应能力差，CCM模式一般用在重载能力，优势为动态负载响应能力好，但是费电；
- RT一般会在进入低功耗模式时会手动软件切换成DCM模式（手动调用DCDC_BootIntoDCM(DCDC) API），而进入CCM模式则是DCDC会在检测到负载（即内核耗电）大于50mA时自动从DCM切换到CCM无需软件配置，但是SDK里面切换CCM时却使用了手动切换，这会导致内核电压在切换时有跌落情况，当跌落到0.915v之下时有潜在风险程序跑飞（下图蓝色为内核电压，在切换时跌落到0.75v，而黄色是某个IO，有异常翻转情况）；
- 建议低功耗切换例程里的lpm.c文件的手动切换CCM API均屏蔽掉，让内部DCDC自动切换。




```
cific.c | specific.c | lpm.c | power_mode_switch_bm.c
    ClockSetToOverDriveRun();
}

void LPM_FullSpeedRun(void)
{
    /* CCM Mode */
    //DCDC_BootIntoCCM(DCDC);
    /* Connect internal the load resistor */
    DCDC->REG1 |= DCDC_REG1_REG_RLOAD_SW_MASK;
    /* Adjust SOC voltage to 1.275V */
    DCDC_AdjustTargetVoltage(DCDC, 0x13, 0x1);
```

RT USB认证问题

- RT的USB是高速USB接口，所以在做USB电路设计时要考虑很多因素，除了layout相关的考量之外，有以下两点需要注意：

（1）首先建议使用boards\evkbimxrt1050\usb_examples\usb_device_hid_mouse例程来做USB测试（官方EVK就是用的该例程做的USB认证），并把工程里usb_device_config.h文件里的compliance test宏打开，这样USB的测试设备才能识别你的USB设备；

usb_device_config.h

```
94
95  /*! @brief Whether device compliance test is enabled. If the macro is enabled,
96      the test mode and CV test macroes will be set.*/
97  #define USB_DEVICE_CONFIG_COMPLIANCE_TEST (1U)
```

（2）USB眼图测试是比较难通过的一点，RT在USB PHY功能模块里提供了如下寄存器来调整USB眼图效果，即调整USB PHY的匹配电阻和电流源大小（高速USB通信是电流驱动的），TXCAL45是调串联45欧电阻，D-CAL是调PHY的电流源大小，USB眼图张开太大，就调大TXCAL_45DP/DN标识那个值，张开太小你就调小，而如果调到最小或者最大之后仍然不满足要求的时候再调整D_CAL。下图给出简单的示例代码；

43.4.2 USB PHY Transmitter Control Register (USBPHYx_TX*n*)

The USB PHY Transmitter Control Register handles the transmit controls.

Address: Base address + 10h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSVD5			USBPHY_TX_EDGECTRL			RSVD2						TXCAL45DP			
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD1				TXCAL45DN				RSVD0				D_CAL			
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1	1

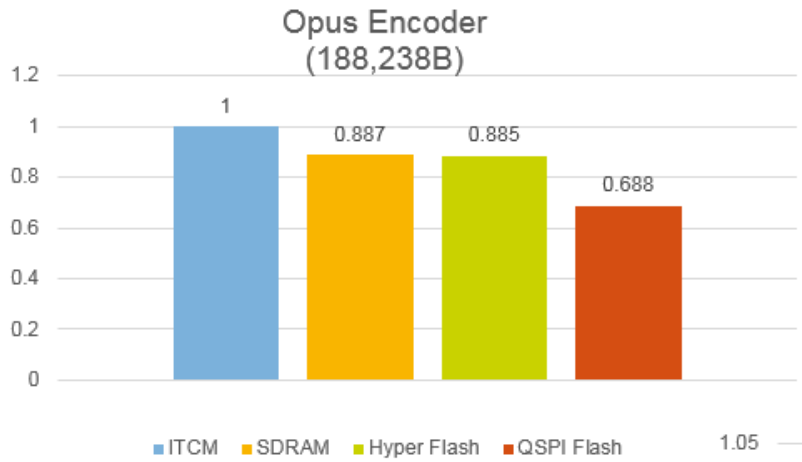
```
USB_DeviceApplicationInit();

//for USB1 Interface
USBPHY1->TX = 0x1C020208;//TXCAL_45DP/DN, D-CAL
USB_ANALOG->INSTANCE[0].CHRG_DETECT = USB_ANALOG_CHRG_DETECT_EN_B_MASK;

//for USB2 Interface
//USBPHY2->TX = 0x1C020208;//TXCAL_45DP/DN, D-CAL
//USB_ANALOG->INSTANCE[1].CHRG_DETECT = USB_ANALOG_CHRG_DETECT_EN_B_MASK;

while (1U)
{
```

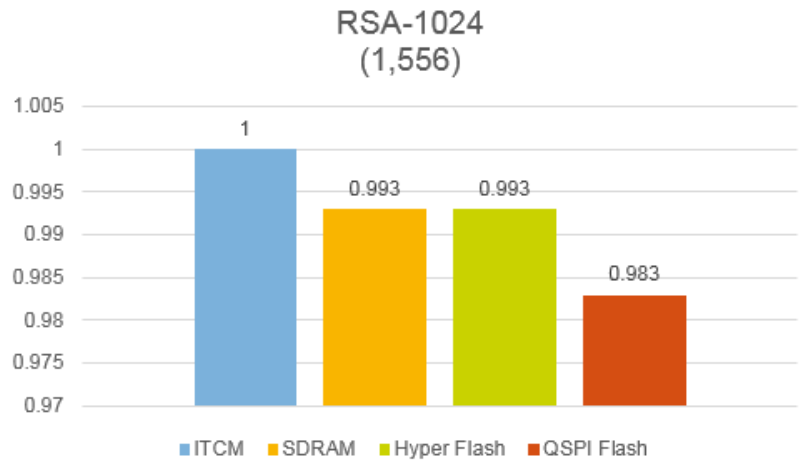
性能对比

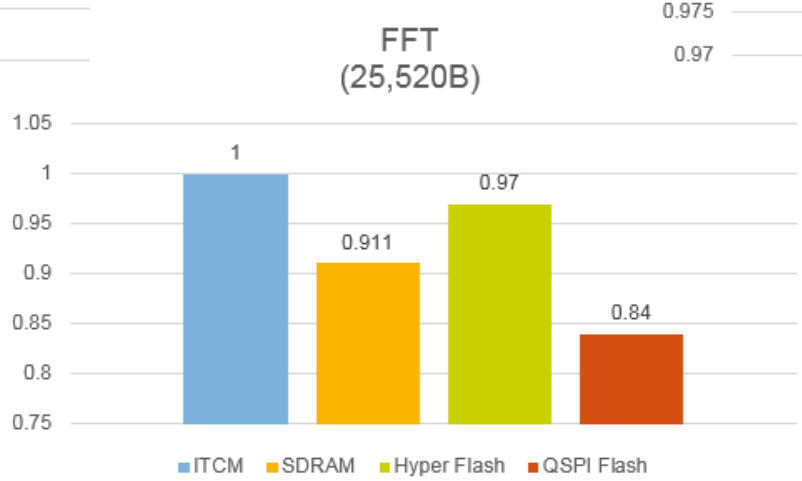
之前在介绍Memory management 的时候，讲到了常用的几种memory，对于多种memory运行代码速度的优先级之前已有提示，具体的指标如下：可以看到TCM>Hyper Flash=SDRAM>QSPI Flash

Test Conditions:

- Core: 600MHz, ITCM: 600MHz, SDRAM: 163MHz SDR, Hyper Flash: 166MHz DDR, QSPI Flash: 127MHz SDR







Using J-Flash to programming QSPI Flash

1. The J-Flash inside the Segger software package is a very useful, friendly and polular tool for product pilot run and even mass production. In the latest version(v6.32 and above), i.MXRT external QSPI Flash programming is supported.

Version V6.32 (2018-04-20)

1. J-Flash: When importing a binary data file, the start address of the the first flash memory bank is set as default start address.
2. DLL: Added API functions: JLINK_ReadMemZonedU32(), JLINK_ReadMemZonedU16(), JLINK_WriteZonedU32(), JLINK_WriteZonedU16()
3. DLL: Added PCode/script file functions JLINK_MEM_Preserve(), JLINK_MEM_Restore(), JLINK_MEM_Fill()
4. DLL: Added command string "MemPreserveOnReset" to specify memory areas that need to be preserved + restored across resets
5. DLL: Added support for accessing memory via different zones/methods (e.g. AHB-AP, APB-AP, ... on Cortex-A/R, to allow live updates). Will be used in future SEGGER Ozon
6. DLL: Debugging on NXP LPC54S0xx devices did not work. Fixed.
7. DLL: Improved debugging on NXP LPC540xx devices.
8. GDBServer: Added support for SEGGER specific GDB protocol extension for streaming trace (qSeggerSTRACE:caps, qSeggerSTRACE:GetInstStats)
9. GDBServer: Improved parsing speed of GDB protocol packets
10. J-Flash Lite: NXP LPC540xx: QSPI could not be erased because banks was not marked as "always present". Fixed. As these devices do not provide internal flash, but QSPI
11. J-Flash: Improved log output during Erase/Program/Read back.
12. J-Flash: Under special circumstances, reading back flash contents could be slowed down by accident (e.g. when reading large parts of the flash with lots of non-programmed
13. J-Flash: When generating a DAT file with big gaps for Flasher stand-alone operation, for parallel CFI NOR flash, it could happen that Flasher failed to flash the file while J-Fla
14. J-Flash: When reading back large flash areas where there were huge non-programmed parts between programmed data areas, J-Flash could enter an endless loop. Fixed.
15. J-Link Commander: NXP LPC540xx: QSPI could not be erased because banks was not marked as "always present". Fixed. As these devices do not provide internal flash, bu
16. J-Link Commander: VTref is now shown with additional information if "fixed VTref" is active.
17. J-Link Configurator: A crash could happen when hitting "Update firmware of selected emulators" but not having any emulators selected. Fixed.
18. Package (Linux): 3rd party plugins may failed to detect certain executables like J-Link GDB Server names. Fixed. (Added symlinks because executable names were changed
19. Package (macOS): 3rd party plugins may failed to detect certain executables like J-Link GDB Server names. Fixed. (Added symlinks because executable names were chang
20. UM08001: Added SEGGER specific GDB protocol extension for streaming trace
21. UM08001: Moved J-Link GDB Server to separate chapter
22. DLL: Added SPI / SPIFI (QSPI) support for Eon EN25QH64 SPI flash.
23. DLL: Added SPI / SPIFI (QSPI) support for Macronix MX25R3235F, MX25L6433F and MX25R4035F SPI flashes.
24. DLL: Added flash programming support for Silicon Labs EFR32MG14PxxxF256, EFR32BG14PxxxF256 and EFR32FG14PxxxF256 series devices.
25. DLL: Added flash programming support for Cypress CY8C4125xxx-PSxxx and CY8C4145xxx-PSxxx series devices.
26. DLL: Added flash programming support for Cypress CYBLE-014008-00, CYBLE-022001-00 and CYBLE-214009-00 series devices.
27. DLL: Added flash programming support for Maxim MAX32552 series devices.
28. DLL: Added flash programming support for Microchip ATSAMHA0E / ATSAMHA0G series devices.
29. DLL: Added flash programming support for Microchip PIC32MX170F512H series devices.
30. DLL: Added flash programming support for NXP LPC804 series devices.
31. DLL: Added flash programming support for ST "STM32L442KC" series devices.
32. DLL: Added flash programming support for Silicon Labs EFM32TG11BxxxF64 and EFM32TG11BxxxF128 series devices.
33. DLL: NXP iMXRT1051 / iMXRT1052: Added HyperFlash flash programming support.
34. DLL: NXP iMXRT1051 / iMXRT1052: Added QSPI flash programming support.
35. DLL: NXP iMXRT1051 / iMXRT1052: Changed device names to more generic ones (MIMXRT1051xxxxA, MIMXRT1052xxxxA, MIMXRT1051xxxxB and MIMXRT1052xxxxB.

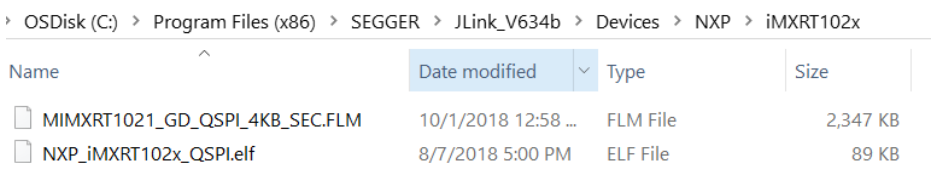
Using J-Flash to programming QSPI Flash

2. The flashloader for RT1050/20/60 could program/Erase many vendors' QSPI Flash, but it doesn't set QE bit, which means chip may boot with failure if we config RT FlexSPI boot FCB information with Quad-line boot by using J-Flash to programming the QSPI Flash for the first time. While Thanks to J-Flash is powerful enough, it supports 3rd party flashloader by following CMSIS standard elf or FLM flash algorithm file. The below link can be taken as reference.

https://wiki.segger.com/Adding_Support_for_New_Devices

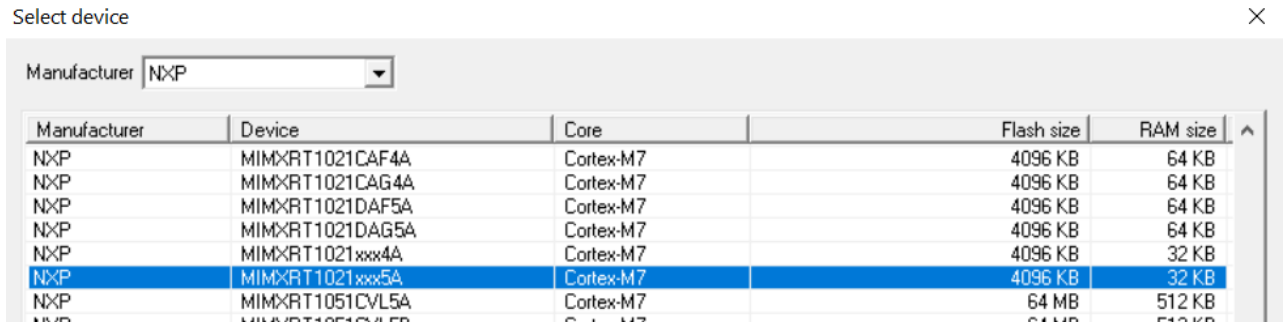
https://github.com/JayHeng/imxrt-tool-flash-algo/tree/master/boards

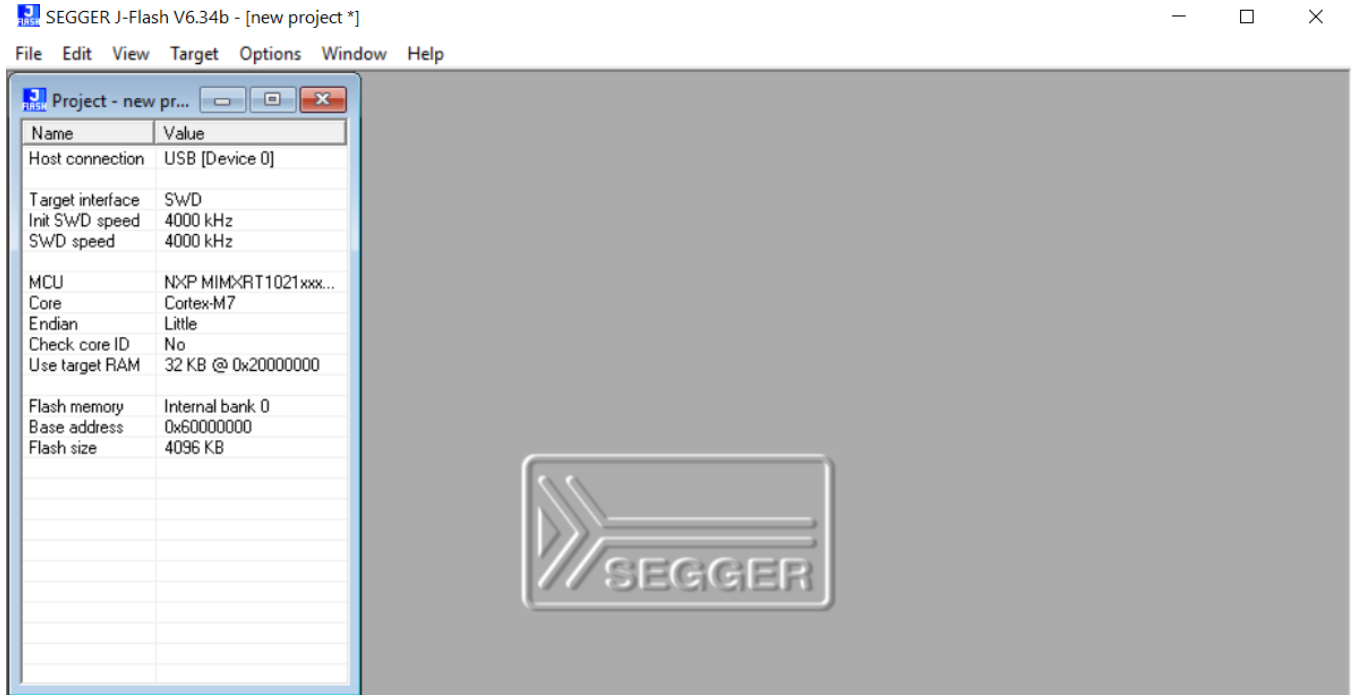
3. Used the above opensource i.mxrt flash algo in github to generate right .FLM file and placed it into Segger-Jlink install path (C:\SEGGER\JLink_V634b\Devices\NXP\iMXRT102x) and modify the JlinkDevices.xml with the new flash algorithm to replace the old one.

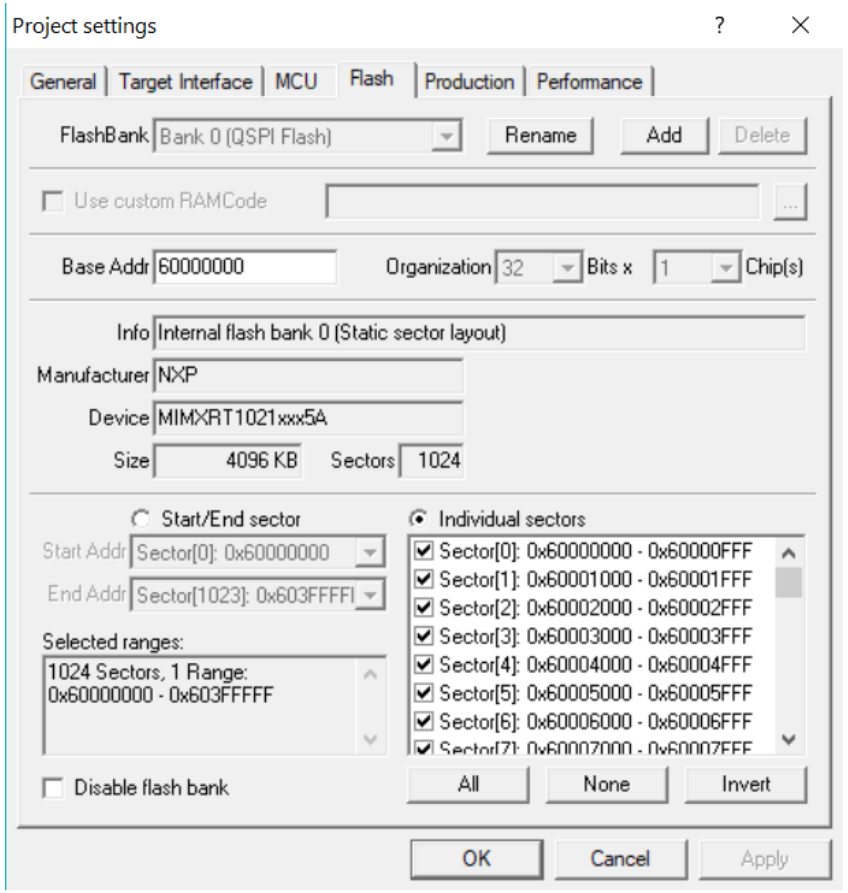


Using J-Flash to programming QSPI Flash

4. Now, everything is ready. Open J-Flash to select i.MXRT1021 and start your journey.





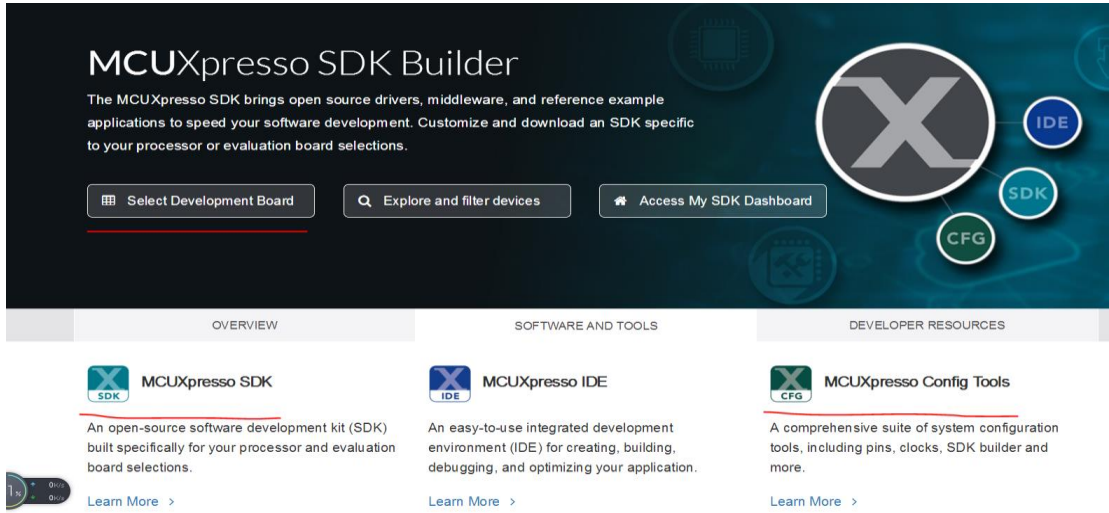


IMX RT 如何使用SDK&PIN CONFIG工具

SDK & Tools

到以下网址可以下载到最新的针对各种RT芯片的SDK，其中包含驱动及USB以太网协议栈等中间件的例程

https://mcuxpresso.nxp.com/en/welcome



MCUXpresso Config Tools可以生成RT的时钟树配置，并与SDK工程进行进行编译

https://www.nxp.com/support/developer-resources/software-development-tools/mcuxpresso-software-and-tools/mcuxpresso-config-tools-pins-clocks-peripherals:MCUXpresso-Config-Tools?tab=Design_Tools_Tab

下载Pins tool for i.MX 可以在硬件设计时就帮助硬件/方案工程师分配硬件模块的管脚，以免在第一版原理图阶段管脚或者模块产生冲突。

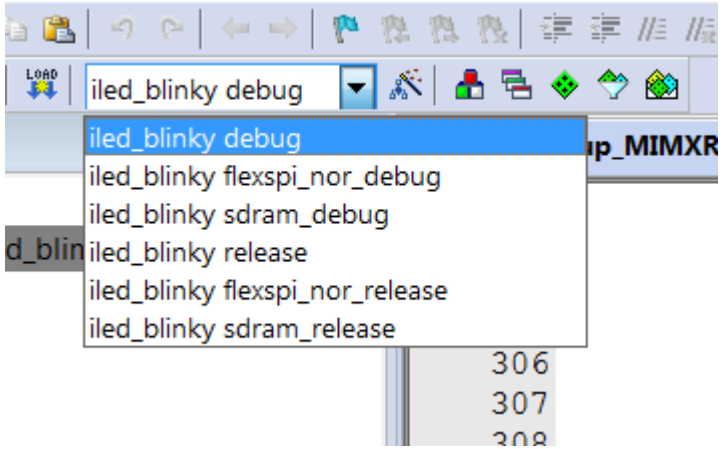
并生成硬件管脚分配的配置程序，直接覆盖到SDK工程目录的pin_mux.c即可完全编译通过

https://www.nxp.com/pages/pins-tool-for-i.mx-application-processors:PINS-TOOL-IMX

SDK & Tools

常见的SDK工程包含的工程配置大概包含以下几类，请在参考这些工程的时候注意甄别：

- Debug：icf将代码定义在ITCM，数据定义在DTCM。如想修改ITCM或者DTCM的大小，在调试时需要修改调试脚本文件，请参考<Redistribute the FlexRAM(ITCM/DTCM/OCRAM) dynamically>章节

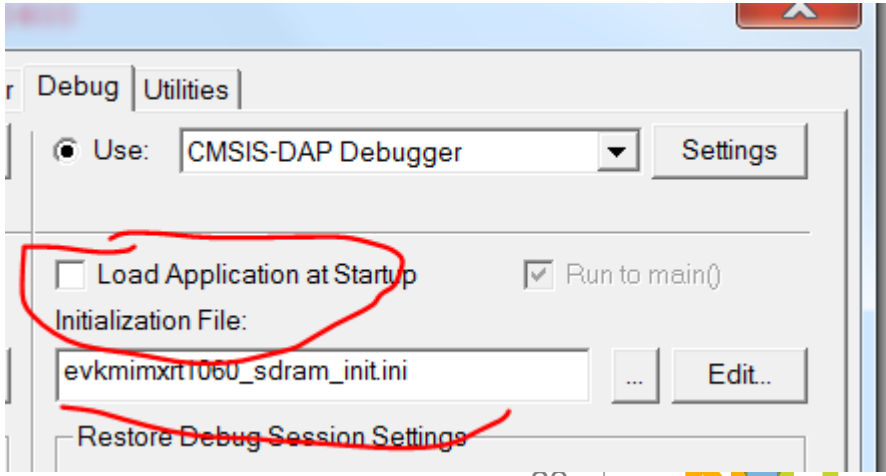


- Flexspi_nor_debug:将代码定义在外部flexspi空间，数据定义在内部ram或者外部sdram。并在此基础上增加启动配置文件XIP信息到程序image。用户自己的板子请检查XIP

- Sdram_debug:代码定义在内部ITCM，数据定义在外部SDRAM，所以调试脚本中有对SDRAM的前期初始化脚本，当调试下载前先初始化外部SDRAM。

- 以上代码参考的时候请注意，一旦修改了默认的SDRAM/FlexRAM空间，请注意检查修改MPU_Setting中，对于RAM的使用是否符合当前应用的需求。请参阅<MPU setting on RT MCU>

- 以上配置凡是使用脚本在下载前修改TCM/SDRAM配置的，务必取消Load Application at Startup这个勾，这样保证先配置，后下载。



SDK & Tools

1.检查一下clock_config.c中

/* Setting the VDD_SOC to 1.25V. It is necessary to config AHB to 600Mhz. */

DCDC->REG3 = (DCDC->REG3 & (~DCDC_REG3_TRG_MASK)) | DCDC_REG3_TRG(0x12);

除了改变从600MHz到528MHz, 之前的SDK默认设置DCDC输出为1.25V.

跑600MHz，1.25V偏低，我们后来建议调高一格DCDC_REG3_TRG(0x13);到1.275V

对于跑528MHz，1.25V合适的。

Table 9. Operating ranges

Parameter Description	Symbol	Operating Conditions	Min	Typ	Max[1]	Unit	Comment
Run Mode	VDD_SOC_IN	Overdrive	1.25	—	1.3	V	—
	VDD_SOC_IN	M7 core at 528 MHz	1.15		1.3	V	—
		M7 core at 132 MHz	1.15	—	1.3		

4-0 TRG	Target value of VDD_SOC, 25 mV each step • 0x0: 0.8V • 0xE: 1.15V • 0x1F:1.575V

SDK & Tools

在使用以太网，SD，USB等例程时需要注意，例程中设置了大的缓冲区，

Flexspi_nor例程，一般将所有数据都放入DTCM；

SDRAM例程一般使用cache control的宏，在读取数据前清空cache；

用户一般会基于自己的工程重新规划RAM资源：需要注意将对应的接收发送缓冲区设置为non-cache(整个放入DTCM)，或者设置如右图的宏，在使用这些buffer之前会清空cache中的原有数据。

另外，对于协议栈代码中含有noncacheable字段的数据，一定将此字段定义在DTCM中

```
RW_m_data m_data_start m_data_size-Stack_Size-Heap_Size { ; RW data
  .ANY (+RW +ZI)
  * (NonCacheable.init)
  * (NonCacheable)
}
;RW_m_data m_data_start m_data_size-Stack_Size-Heap_Size { ; RW data
;  .ANY (+RW +ZI)
;}
;RW_m_data2 m_data2_start m_data2_size-Stack_Size-Heap_Size { ; RW data
;  * (NonCacheable.init)
;  * (NonCacheable)
;}
ARM_LIB_HEAP +0 EMPTY Heap_Size {    ; Heap region growing up
}
;ARM_LIB_HEAP m_data2_start EMPTY Heap_Size{    ; Heap region growing up
;}
ARM_LIB_STACK m_data2_start+m_data2_size EMPTY -Stack_Size { ; Stack region growing down
}
```

IMX RT Learning materials

Valued Application Notes

1. AN12183-How to Enable Debugging for FLEXSPI NOR Flash
2. AN12108-How to Enable Boot from QSPI Flash
3. AN12042-Using the i.MXRT L1 Cache
4. AN12077-Using the i.MX RT FlexRAM
5. AN12085-How to use iMXRT Low Power feature
6. AN12238-i.MXRT Flashloader use case(including FUSE PROGRAM steps)

Valued Tools

1. Flashloader i.MX-RT1050 & Flashloader i.MX-RT1020

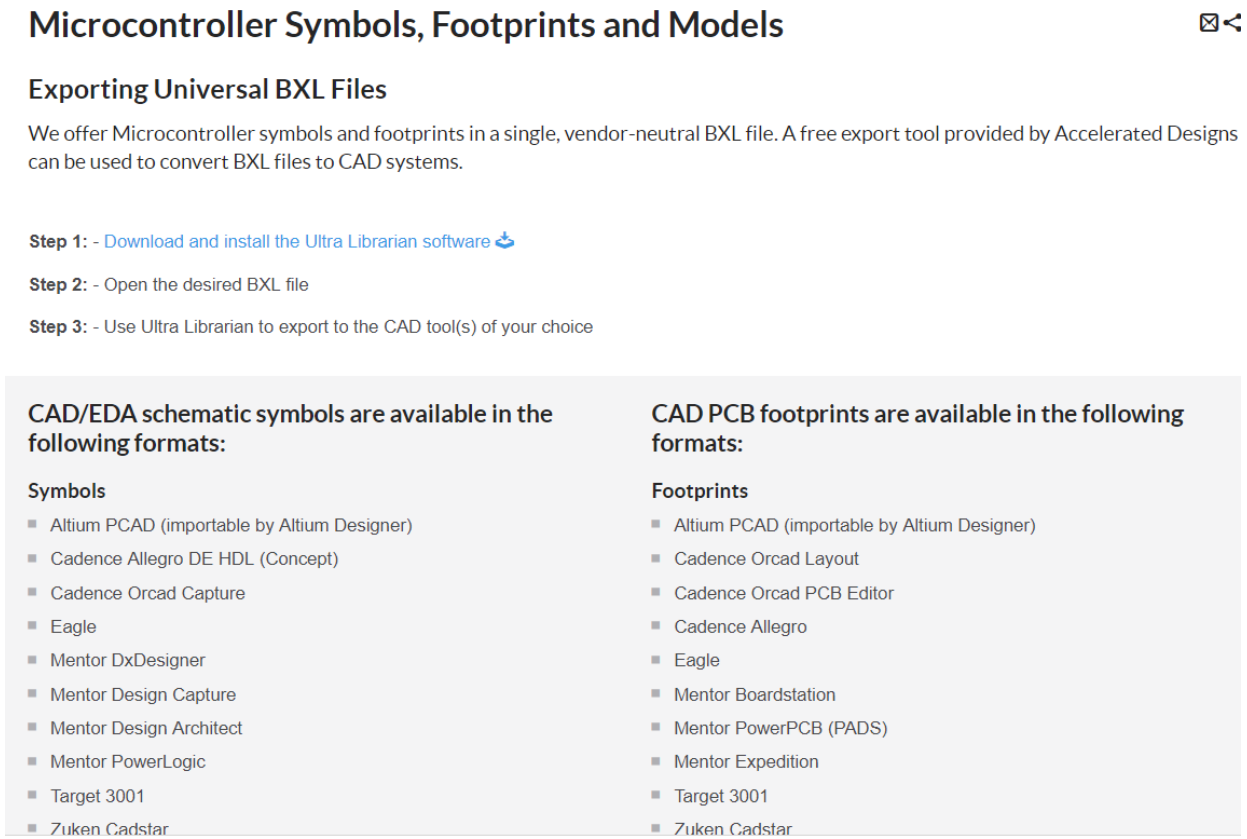
- The flashloader is a companion tool to the i.MX Boot ROM and offers a solution for generating bootable images and programming boot images into boot devices.
- Supports programming of boot devices:
  - QuadSPI NOR/Octal Flash / HyperFLASH
  - Serial NAND
  - eMMC
  - SD
  - Parallel NOR
  - SLC raw NAND
  - SPI NOR/EEPROM

Valued Tools

2. Chip package export tool

- https://www.nxp.com/support/developer-resources/models-and-test-data/microcontroller-symbols-footprints-and-models:MCUCAD?fsrch=1&sr=1&pageNum=1#

- A good efficiency tool to export MCU's schematic and PCB footprints package to common CAD/EDA tool, like Altium Designer and Cadence.



Microcontroller Symbols, Footprints and Models

Exporting Universal BXL Files

We offer Microcontroller symbols and footprints in a single, vendor-neutral BXL file. A free export tool provided by Accelerated Designs can be used to convert BXL files to CAD systems.

Step 1: - Download and install the Ultra Librarian software

Step 2: - Open the desired BXL file

Step 3: - Use Ultra Librarian to export to the CAD tool(s) of your choice

CAD/EDA schematic symbols are available in the following formats:

Symbols

- Altium PCAD (importable by Altium Designer)
- Cadence Allegro DE HDL (Concept)
- Cadence Orcad Capture
- Eagle
- Mentor DxDesigner
- Mentor Design Capture
- Mentor Design Architect
- Mentor PowerLogic
- Target 3001
- Zuken Cadstar

CAD PCB footprints are available in the following formats:

Footprints

- Altium PCAD (importable by Altium Designer)
- Cadence Orcad Layout
- Cadence Orcad PCB Editor
- Cadence Allegro
- Eagle
- Mentor Boardstation
- Mentor PowerPCB (PADS)
- Mentor Expedition
- Target 3001
- Zuken Cadstar

Valued Tools

3. NXP-MCUBootUtility and NXP-MCUBootFlasher

- MCUBootUtility for i.MXRT debug and test

https://github.com/JayHeng/NXP-MCUBootUtility

- MCUBootFlasher for i.MXRT MP programming

https://github.com/JayHeng/NXP-MCUBootFlasher

Good Community and BBS

1. NXP Community
https://Community.nxp.com/welcome

2. 野火RT1050板块
http://www.firebbs.cn/forum-RT1050-1.html

3. 正点原子RT1050板块
http://www.openedv.com/forum-252-1.html

4. NXP官方技术分享微信公众号 ◄

SECURE CONNECTIONS
FOR A SMARTER WORLD