KW36 - 32kHz RTC 外部振荡器的微调调节

USL：https://community.nxp.com/docs/DOC-342672

引言

FRDM-KW36 包含带有 32 kHz 晶体振荡器的 RTC 模块。RTC 模块以极低功耗模式运行并为 MCU 提供 32 kHz 时钟源。该振荡器包括一组可编程调节的负载电容 C_{LOAD}，改变这些负载电容的值可以调整振荡器提供的频率。

此可配置电容的范围为 0 pF（禁用电容器组）至 30 pF，步长为 2 pF。 这些值是通过组合启用的电容器获得的。可用值为 2 pF，4 pF，8 pF 和 16 pF。这四个数值可以任意组合。如果外部电容可用，建议禁用这些内部电容器（将 RTC 控制寄存器 SFR 中的 SC2P，SC4P，SCS8 和 SC16 位设置为 0）。

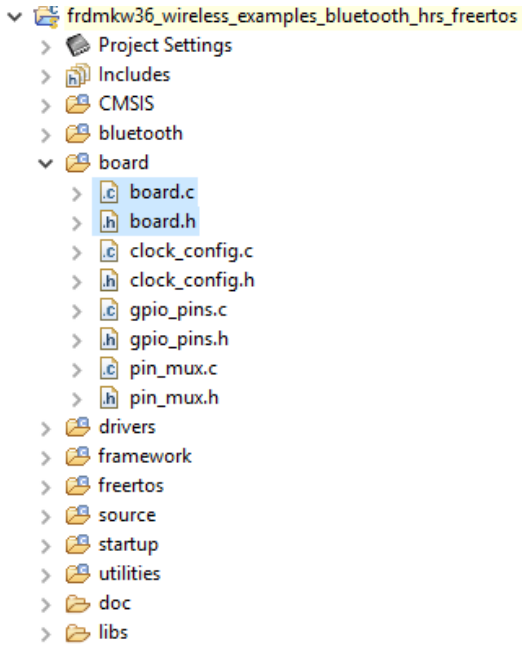
要调整振荡器提供的频率，必须首先能够测量该频率。最好使用频率计数器测了频率，因为它提供了比示波器更精确的测量。另外还需要 KW36 通过 IO 输出振荡器频率。要输出振荡器频率，以任意一个低功耗蓝牙演示应用程序为例，执行以下操作：

调整频率示例

本示例将利用低功耗蓝牙演示应用程序的心率传感器演示（freertos 版本），并假定开发人员具有从 SDK 到 IDE 导入或打开项目的知识。

1. 从 SDK 中打开或克隆“心率传感器”项目。

2. 在工作区的 board 文件夹中找到 board.c 和 board.h 文件。



3. 如下图所示在 board.h 文件中声明一个 void 函数。该函数将是为了把 RTC 时钟多路复用到 PTB3，以使其能够输出 32kHz 频率用于测量。

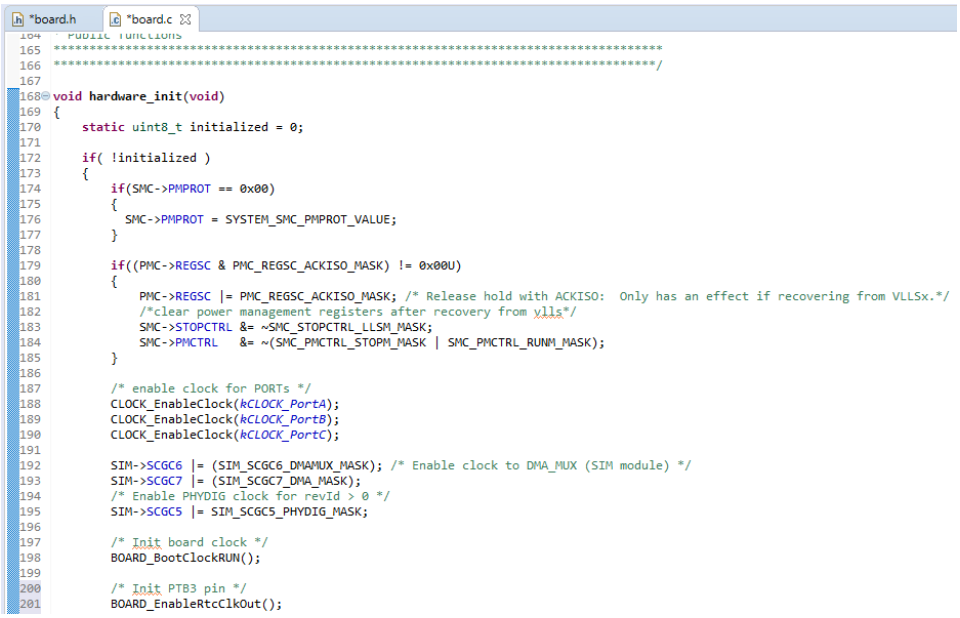
```
/* Function to mux PTB3 to RTC_CLKOUT */
void BOARD_EnableRtcClkOut (void);
```

4. 如下所示在 board.c 文件中添加 BOARD_EnableRtcClkOut 函数。

5.

```
void BOARD_EnableRtcClkOut(void)
{
/* Enable PORTB clock gating */
CLOCK_EnableClock(kCLOCK_PortB);
/* Mux the RTC_CLKOUT to PTB3 */
PORT_SetPinMux(PORTB, 3u, kPORT_MuxAlt7);
/* Select the 32kHz reference for RTC_CLKOUT signal */
SIM->SOPT1 |= SIM_SOPT1_OSC32KOUT(1);
 }
```

6. 在 hardware_init 函数中（board.c 文件），在调用 BOARD_BootClockRUN 函数之后立即调用 BOARD_EnableRtcClkOut 函数。



7. 在工作区的 board 文件夹中找到 clock_config.c 文件。

8. 在文件顶部添加以下定义。

```
#define RTC_OSC_CAP_LOAD_0 0x0U /*!< RTC oscillator, capacitance 0pF */
#define RTC_OSC_CAP_LOAD_2 0x2000U /*!< RTC oscillator, capacitance 2pF */
#define RTC_OSC_CAP_LOAD_4 0x1000U /*!< RTC oscillator, capacitance 4pF */
#define RTC_OSC_CAP_LOAD_6 0x3000U /*!< RTC oscillator, capacitance 6pF */
#define RTC_OSC_CAP_LOAD_8 0x800U /*!< RTC oscillator, capacitance 8pF */
#define RTC_OSC_CAP_LOAD_10 0x2800U /*!< RTC oscillator, capacitance 10pF *
/
#define RTC_OSC_CAP_LOAD_12 0x1800U /*!< RTC oscillator, capacitance 12pF *
/
#define RTC_OSC_CAP_LOAD_14 0x3800U /*!< RTC oscillator, capacitance 14pF *
/
#define RTC_OSC_CAP_LOAD_16 0x400U /*!< RTC oscillator, capacitance 16pF */
#define RTC_OSC_CAP_LOAD_18 0x2400U /*!< RTC oscillator, capacitance 18pF *
/
#define RTC_OSC_CAP_LOAD_20 0x1400U /*!< RTC oscillator, capacitance 20pF *
/
#define RTC_OSC_CAP_LOAD_22 0x3400U /*!< RTC oscillator, capacitance 22pF *
/
#define RTC_OSC_CAP_LOAD_24 0xC00U /*!< RTC oscillator, capacitance 24pF */
#define RTC_OSC_CAP_LOAD_26 0x2C00U /*!< RTC oscillator, capacitance 26pF *
/
```

```
#define RTC_OSC_CAP_LOAD_28 0x1C00U /*!< RTC oscillator, capacitance 28pF *
/
#define RTC_OSC_CAP_LOAD_30 0x3C00U /*!< RTC oscillator, capacitance 30pF *
/
```

9. 在 BOARD_BootClockRUN 函数内（也在 clock_config.c 文件中）找到对函数 CLOCK_CONFIG_EnableRtcOsc 的调用，然后通过上述任意定义来设置函数入参。
10. 最后，在项目源文件夹中的“app_preinclude.h”文件中禁用低功耗选项和 LED Support：

```
#define cPWR_UsePowerDownMode 0
#define gLEDSupported_d 0
```

此时，可以用频率计数器测量 PTB3 输出的频率，并进行频率调整。每次对电路板进行编程时，都需要执行 POR 以获得正确的测量值。下表是从 FRDM-KW36 板 rev B 获得的，可用作调整频率的参考。 请注意，电容不仅由启用的内部电容组成，还包括封装、焊线、焊垫和 PCB 走线中的寄生电容。因此，尽管下面给出的参考测量值应接近实际值，但您还应该在电路板上进行测量，以确保频率是专门针对您的电路板和布局进行调整的。

启用的电容器	CLOAD	电容定义	频率
-	0pF	RTC_OSC_CAP_LOAD_0 (bank disabled)	32772.980Hz
SC2P	2pF	RTC_OSC_CAP_LOAD_2	32771.330Hz
SC4P	4pF	RTC_OSC_CAP_LOAD_4	32770.050Hz
SC2P, SC4P	6pF	RTC_OSC_CAP_LOAD_6	32769.122Hz
SC8P	8pF	RTC_OSC_CAP_LOAD_8	32768.289Hz
SC2P, SC8P	10pF	RTC_OSC_CAP_LOAD_10	32767.701Hz
SC4P, SC8P	12pF	RTC_OSC_CAP_LOAD_12	32767.182Hz
SC2P, SC4P, SC8P	14pF	RTC_OSC_CAP_LOAD_14	32766.766Hz
SC16P	16pF	RTC_OSC_CAP_LOAD_16	32766.338Hz
SC2P, SC16P	18pF	RTC_OSC_CAP_LOAD_18	32766.038Hz
SC4P, SC16P	20pF	RTC_OSC_CAP_LOAD_20	32765.762Hz
SC2P, SC4P, SC16P	22pF	RTC_OSC_CAP_LOAD_22	32765.532Hz
SC8P, SC16P	24pF	RTC_OSC_CAP_LOAD_24	32765.297Hz
SC2P, SC8P, SC16P	26pF	RTC_OSC_CAP_LOAD_26	32765.117Hz

启用的电容器	CLOAD	电容定义	频率
SC4P, SC8P, SC16P	28pF	RTC_OSC_CAP_LOAD_28	32764.940Hz
SC2P, SC4P, SC8P, SC16P	30pF	RTC_OSC_CAP_LOAD_30	32764.764Hz