简介

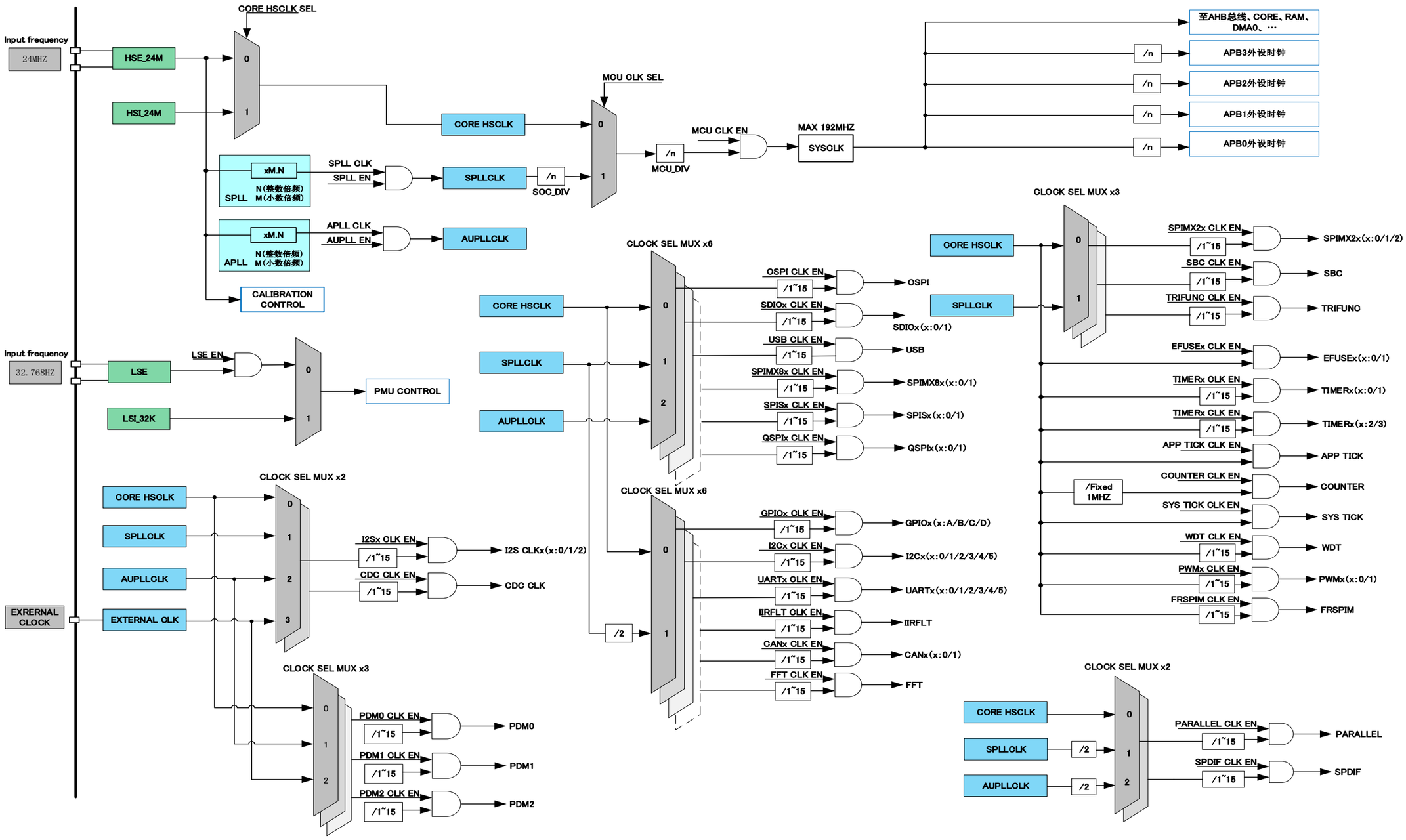
本应用手册具体介绍了在 Fr30xx 系列芯片中，使用时钟配置时的注意事项与使用方法。

旨在减少研发工程师的开发时间，提高效率，缩短项目周期。

目录

1. 时钟图

FR30xx 的时钟源介绍请参考《FR30xx Specification》系统控制单元中的时钟控制章节。



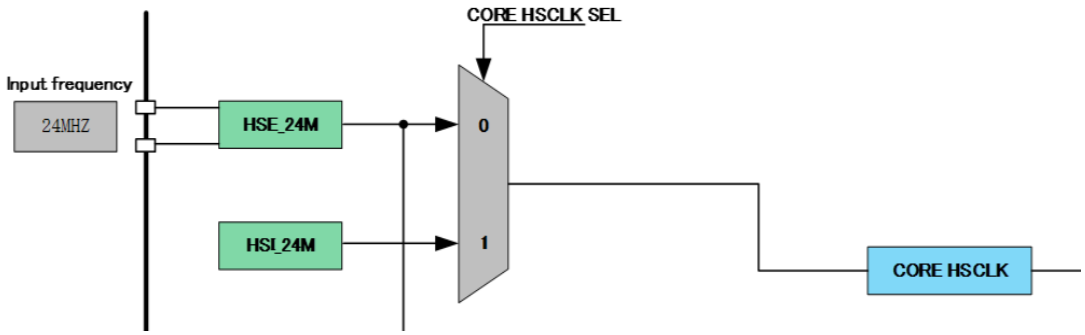
2. 系统时钟配置

初始化系统时钟时需调用 void system_clock_config(void)，用户根据实际需求设置 CORE HSCLK 的源选择、SPLL 和 AUPLL 的整数与小数倍频、MCU CLK 的源选择、SOC 分频、MCUCLK 分频、APB0/1/2/3 分频。

1. CORE HSCLK 的源选择：

可选 CORE_HSCLK_SEL_HES（HSE_24M）或 CORE_HSCLK_SEL_HIS(HSI_24M)。

```
System_ClkConfig_t ClkConfig;

/* CORE HSCLK Config */
ClkConfig.CORE_HSCLK_CFG.CORE_HSCLK_Source = CORE_HSCLK_SEL_HES;
```



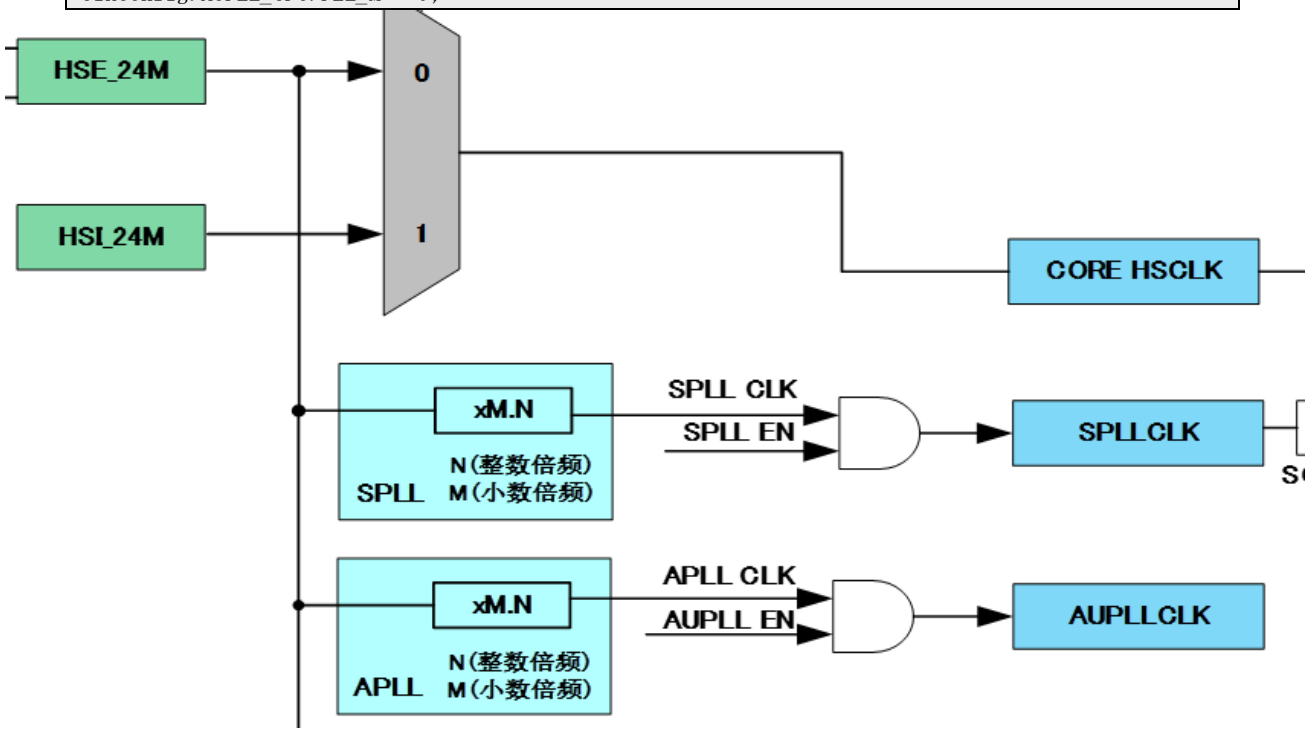
2. SPLL 和 AUPLL 整数倍频与小数倍频的配置

SOC 内部有两个 PLL 锁相环，分别是 SPLL 和 AUPLL。SPLL（System PLL）产生的时钟主要为系统或外设提供时钟源，AUPLL（Audio PLL）产生的时钟主要为音频外设提供时钟源。

SPLL 和 AUPLL 通过锁相环倍频输出时钟。整数与小数倍频的参数根据公式 PLL clock = HSE_VALUE*N + (HSE_VALUE/65535)*M 进行计算。其中 HSE_VALUE 为固定的 24M，N 为整数倍频，M 为小数倍频。

```
/* PLL clock = HSE_VALUE*N + (HSE_VALUE/65535)*M */
/* SPLL CLK Config */
ClkConfig.SPLL_CFG.PowerEn = PLL_POWER_ENABLE;
ClkConfig.SPLL_CFG.PLL_N = 8;
ClkConfig.SPLL_CFG.PLL_M = 0;
```

```
/* PLL clock = HSE_VALUE*N + (HSE_VALUE/65535)*M */
/* AUPLL CLK Config */
ClkConfig.AUPLL_CFG.PowerEn = PLL_POWER_ENABLE;
ClkConfig.AUPLL_CFG.PLL_N = 8;
ClkConfig.AUPLL_CFG.PLL_M = 0;
```

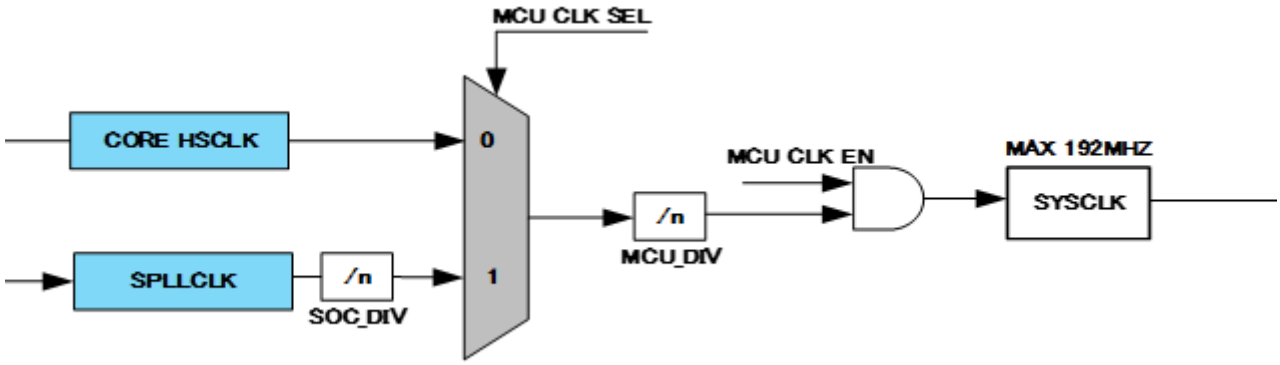


注：由于锁相环特性，在初始化时 PLL_N 整数倍频建议 >= 8，此时 PLL 输出较为稳定。如果需要锁相环产生较低的时钟速率，可以使用外设的分频器或系统时钟的分频器对 PLL 时钟分频。

3. MCU CLK 的源选择、SOC 分频、MCUCLK 分频：

MCU_Clock_Source 可选的时钟源配置分别为 MCU_CLK_SEL_CORE_HSCLK（CORE HSCLK）、MCU_CLK_SEL_SPLL_CLK（SPLL CLK），其中对于锁相环倍频而来的 SPLL CLK 可以设置 1 到 15 范围内的可编程系数分频。在完成 MCU CLK 时钟配置后即可得到所需 MCU CLK。最后，MCU CLK 同样可以进行 1 到 15 范围内的可编程系数分频。

```
ClkConfig.MCU_Clock_Source = MCU_CLK_SEL_CORE_HSCLK;

ClkConfig.SOC_DIV  = 1;    /* This parameter is valid when MCU_Clock_Source ==
MCU_CLK_SEL_SPLL_CLK can be a value 1~15 */ */

ClkConfig.MCU_DIV  = 1; /* This parameter can be a value 1~15 */
```



4. APB0/1/2/3 分频：

MCU CLK 完成配置后得到 SYSCLK，建议最高频率为 192MHZ。确定好 SYSCLK 之后需要配置 APB0/1/2/3 的时钟分频。对 SYSCLK 进行 1 到 15 范围内的分频后获取，如下图所示。

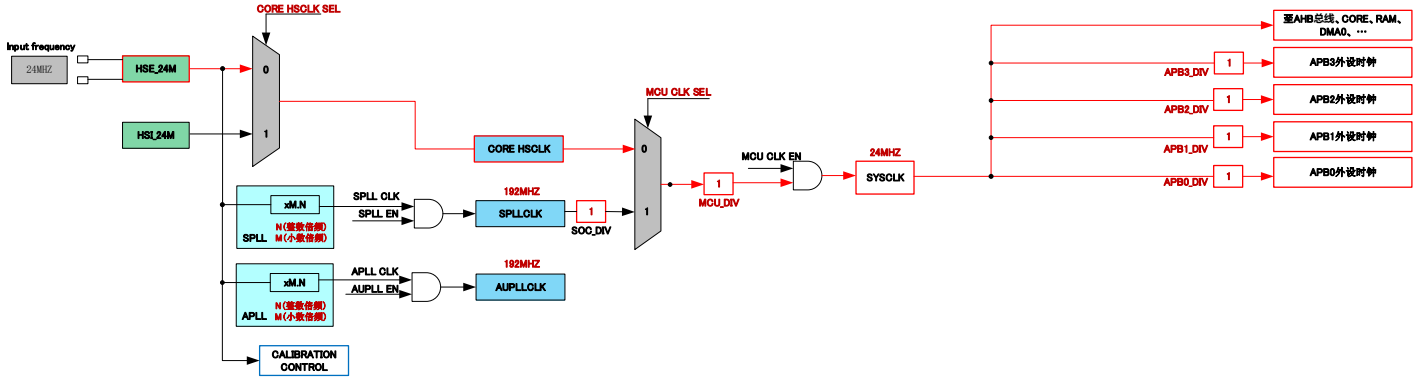
```
ClkConfig.APB0_DIV = 1; /* This parameter can be a value 1~15 */
ClkConfig.APB1_DIV = 1; /* This parameter can be a value 1~15 */
ClkConfig.APB2_DIV = 1; /* This parameter can be a value 1~15 */
ClkConfig.APB3_DIV = 1; /* This parameter can be a value 1~15 */
```

3. 系统时钟配置例程

下图为一个系统时钟配置的完整案列，其中 CORE HSCLK 的时钟源源选择为 HSE_24M，SPLL 与 AUPLL 使能，整数倍频与小数分频分别配置成 8、0。MCU CLK 选择 COREHSCLK 作为时钟源，MCU、APB0/1/2/3 的时钟分频默认 1。由于 MCU CLK 时钟源未选择 SPLL CLK，那么此时就不需要关心 SOC DIV。

```
void system_clock_config(void)
{
    System_ClkConfig_t ClkConfig;

    /* CORE HSCLK Config */
    ClkConfig.CORE_HSCLK_CFG.CORE_HSCLK_Source = CORE_HSCLK_SEL_HES;
    /* PLL clock = HSE_VALUE*N + (HSE_VALUE/65535)*M */
    /* SPLL CLK Config */
    ClkConfig.SPLL_CFG.PowerEn = PLL_POWER_ENABLE;
    ClkConfig.SPLL_CFG.PLL_N = 8;
    ClkConfig.SPLL_CFG.PLL_M = 0;
    /* PLL clock = HSE_VALUE*N + (HSE_VALUE/65535)*M */
    /* AUPLL CLK Config */
    ClkConfig.AUPLL_CFG.PowerEn = PLL_POWER_ENABLE;
    ClkConfig.AUPLL_CFG.PLL_N = 8;
    ClkConfig.AUPLL_CFG.PLL_M = 0;

    System_CORE_HSCLK_config(&ClkConfig.CORE_HSCLK_CFG);
    if (System_SPLL_config(&ClkConfig.SPLL_CFG,200) == -1)
        while(1);
    if (System_AUPLL_config(&ClkConfig.AUPLL_CFG,200) == -1)
        while(1);

    ClkConfig.MCU_Clock_Source = MCU_CLK_SEL_CORE_HSCLK;
    ClkConfig.SOC_DIV  = 1;    /* This parameter is valid when MCU_Clock_Source == MCU_CLK_SEL_SPLL_CLK */
    ClkConfig.MCU_DIV  = 1;
    ClkConfig.APB0_DIV = 1;
    ClkConfig.APB1_DIV = 1;
    ClkConfig.APB2_DIV = 1;
    ClkConfig.APB3_DIV = 1;

    System_MCU_clock_Config(&ClkConfig);
}
```

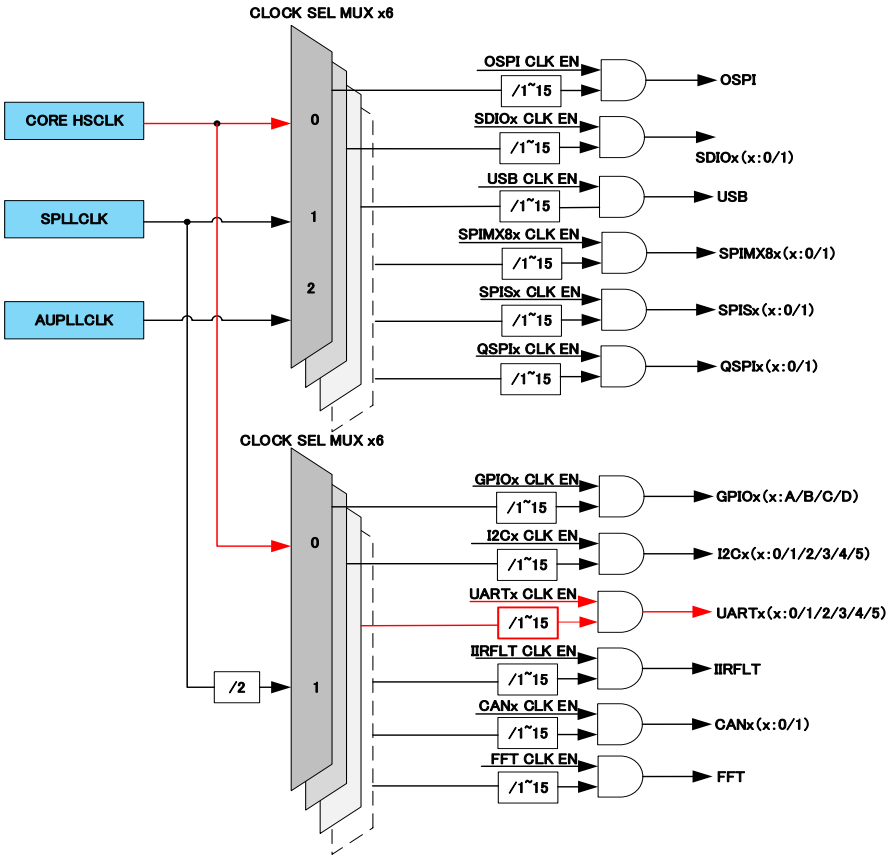

结合上文的案列，可以得到一个完整的系统时钟配置图，下图将用红色实线指示时钟选择路线，用红色字体代表需要完成的配置。



4. 外设时钟配置

FR303x 的外设时钟配置可选多个时钟源，下文将选取 UARTx 时钟源选择进行演示，结合下图，可以看到 UARTx 的时钟源有两个，分别是 CORE HSCLK 和 SPLL CLK。

配置 UART 前需要打开 UART 的时钟使能，同时可以配置 1 到 15 范围内的可编程系数分频。



相关的 API 接口可以在 system._fr303x.h 中找到，UART 时钟源选择后 UART0/1/2/3/4/5 共用同一个配置，但是时钟使能需要单独配置。

```
#define __SYSTEM_UART_CLK_SELECT_COREH()          (SYSTEM->ClockSEL0.UART_CLK_SEL = 0)
#define __SYSTEM_UART_CLK_SELECT_SPLL()           (SYSTEM->ClockSEL0.UART_CLK_SEL = 1)
#define __SYSTEM_UART_CLK_DIV(__CLKDIV__)         (SYSTEM->BlockClockDIV1.UART_CLK_DIV = __CLKDIV__ - 1)

#define __SYSTEM_UART0_CLK_ENABLE()               (SYSTEM->ClockEnable5.UART0_CLKEN = 1)
#define __SYSTEM_UART1_CLK_ENABLE()               (SYSTEM->ClockEnable5.UART1_CLKEN = 1)
#define __SYSTEM_UART2_CLK_ENABLE()               (SYSTEM->ClockEnable5.UART2_CLKEN = 1)
#define __SYSTEM_UART3_CLK_ENABLE()               do {    \
                                                      SYSTEM->ClockEnable5.UART2_CLKEN = 1;   \
                                                      SYSTEM->ClockEnable5.UART3_CLKEN = 1;   \
                                                  } while(0)
#define __SYSTEM_UART4_CLK_ENABLE()               (SYSTEM->ClockEnable5.UART4_CLKEN = 1)
```

```
#define __SYSTEM_UART5_CLK_ENABLE()                    (SYSTEM->ClockEnable5.UART5_CLKEN = 1)
```

5. 变更历史

版本号	日期	更新内容
V1.0	2023.7.27	首版

6. 联系信息

公司：上海富芮坤微电子有限公司

地址：中国(上海)自由贸易试验区碧波路 912 弄 8 号 501-A 室

电话：+86-21-5027-0080

Website: www.freqchip.com

Sales Email: sales@freqchip.com