

使用指南

关于如何使用LSE时钟安全系统提高时钟系统鲁棒性使用指南

简介

此文档的目的在于让使用者了解 N32L43x、N32L40x、N32G43x 系列 LSE 时钟的安全监控功能，提高时钟系统鲁棒性,以及提高方案安全性能,降低开发时间和难度。

目录

1	介绍.....	1
1.1	概述	1
1.2	LSE时钟简介	1
1.3	LSI时钟简介	1
1.4	LSE时钟安全系统（LSECSS）简介	1
1.5	适用性	1
2	硬件环境.....	1
2.1	例程功能	1
2.2	硬件平台	2
3	例程讲解.....	3
3.1	例程流程	3
3.2	例程分析	4
3.2.1	USART1 日志串口初始化	4
3.2.2	使能LSE	5
3.2.3	使能LSI	5
3.2.4	使能LSE-CSS监控	6
3.2.5	进入Stop2	7
3.2.6	初始化LPTIM和LPUART	8
4	使用指南.....	9
4.1	复位MCU	9
4.2	产生LSE故障	10
4.3	LPTIMER切换时钟源	11
4.4	MCU从Stop2中被唤醒	12
4.5	MCU从进入Stop2休眠模式	13
4.6	MCU周期性唤醒	14
4.7	LSE恢复	15
4.8	LPTIMER切换时钟源为LSE	16
5	历史版本.....	17
6	声 明.....	18

1 介绍

1.1 概述

在一些应用场景可能会遇到 LSE 发生故障的情况,在此提出一种可利用 LSE-CSS 监控 LSE 故障,当故障发生时可将系统时钟由 LSE 切换到 LSI 的方法,避免 LSE 故障导致系统停止运行。

1.2 LSE 时钟简介

LSE 晶体是一个 32.768KHz 的低速外部晶体或陶瓷谐振器。它为实时时钟或者其他定时功能提供一个低功耗且精确的时钟源。LSE 晶体通过在低功耗域控制寄存器 (RCC_LDCTRL) 里的 LSEEN 位启动和关闭。在低功耗域控制寄存器 (RCC_LDCTRL) 里的 LSERD 指示 LSE 晶体振荡是否稳定。在启动阶段,直到这个位被硬件置 1 后, LSE 时钟信号才被释放出来。如果在时钟中断寄存器里被允许,可产生中断请求。

1.3 LSI 时钟简介

LSI RC 可以在 STOP2 和 STANDBY 模块下为 IWDG 和 AWU 提供时钟。LSI 时钟频率大约 40KHz。LSI RC 可以通过控制/状态寄存器 (RCC_CTRLSTS) 里的 LSIEN 位来启动或关闭。在控制/状态寄存器 (RCC_CTRLSTS) 里的 LSIRD 位指示低速内部振荡器是否稳定。在启动阶段,直到这个位被硬件设置为 1 后,此时钟才被释放。如果在时钟中断寄存器 (RCC_CLKINT) 里被允许,将产生 LSI 中断申请。

1.4 LSE 时钟安全系统 (LSECSS) 简介

通过使能低功耗域控制寄存器 (RCC_LDCTRL) 的 LSECLKSEN 位,来激活 LSE 时钟安全系统。硬件复位或者 RTC 软件复位或者检测到 LSE 故障后可以清除 LSECLKSEN 位。当 LSE 和 LSI 使能并就绪,在配置 RTCSEL 选择 RTC 时钟源后必须使能 LSECLKSEN 位。如果检测到 LSE 故障,则不会再向 RTC 提供 LSE,但硬件不会修改 RTCSEL 位来切换 RTC 时钟源。在 STANDBY 模式下, LSE 时钟故障会触发唤醒。在其他模式下可以产生中断来唤醒,再由软件清除 LSECLKSEN 位并关闭 LSE,并更改 RTC 的时钟源等其他措施来确保应用的安全。LSE 振荡器的频率必须高于 30KHz,以免发生 LSECSS 误检。

1.5 适用性

此 Demo 仅适用于 N32L43x、N32L40x、N32G43x 系列 MCU,支持 KEIL5 平台。

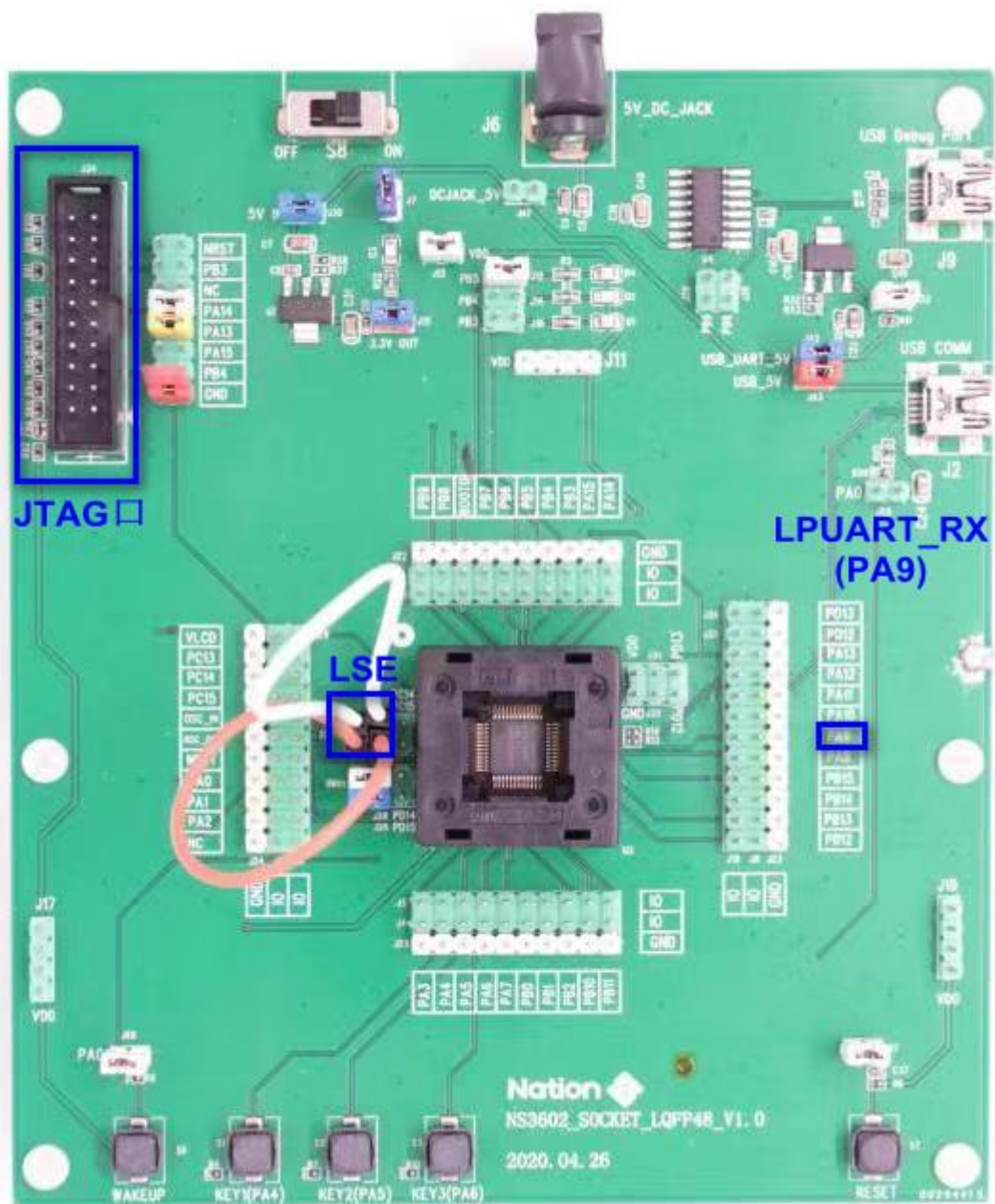
[SDK-VER 1.1.0]

Release Date: 2021-11-30

2 硬件环境

2.1 例程功能

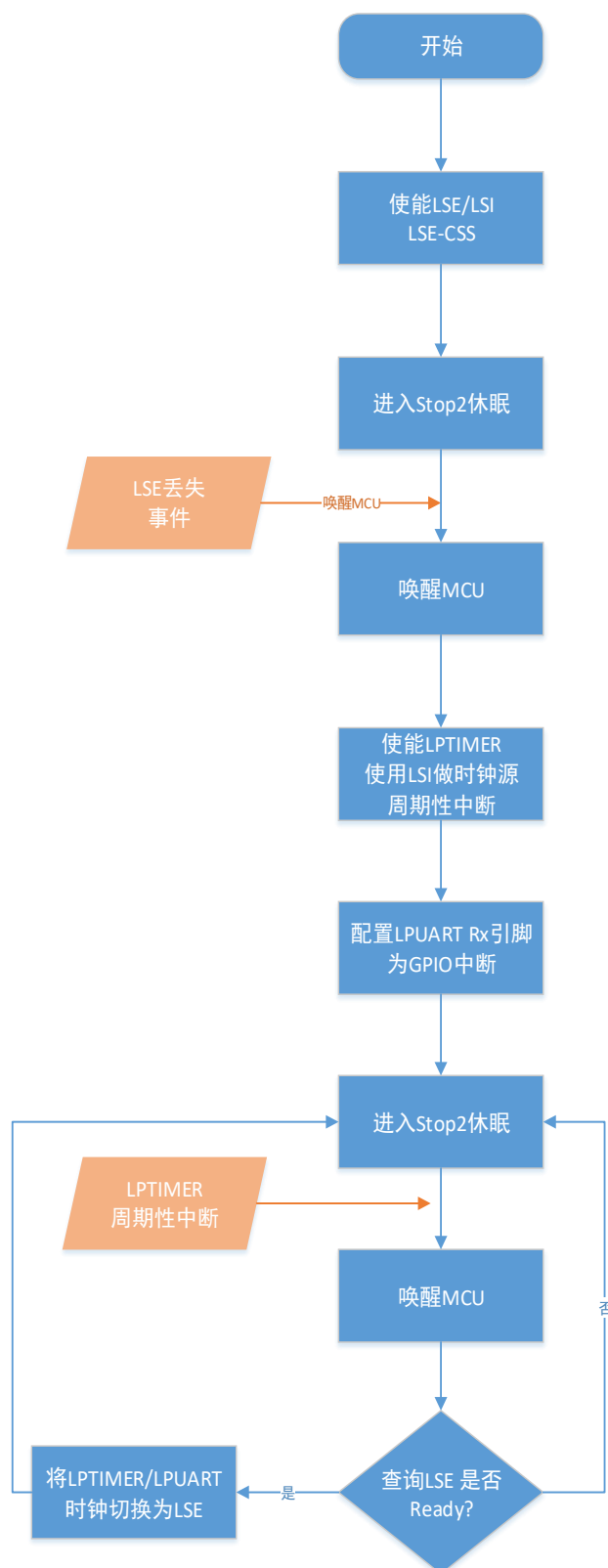
该例程主要给开发者演示,当 LSE 发生故障时,系统软件如何将外设时钟源切换到 LSI,系统监控 LSE 时钟,等待 LSE 恢复后,再将外设时钟源由 LSI 切换到 LSE 的过程。



No.	资源	说明	备注
1	NS3602_SOCKET_LQFP48_V1.0	国民 LQFP48 封装测试座	
2	N32G435CBL7	MCU	

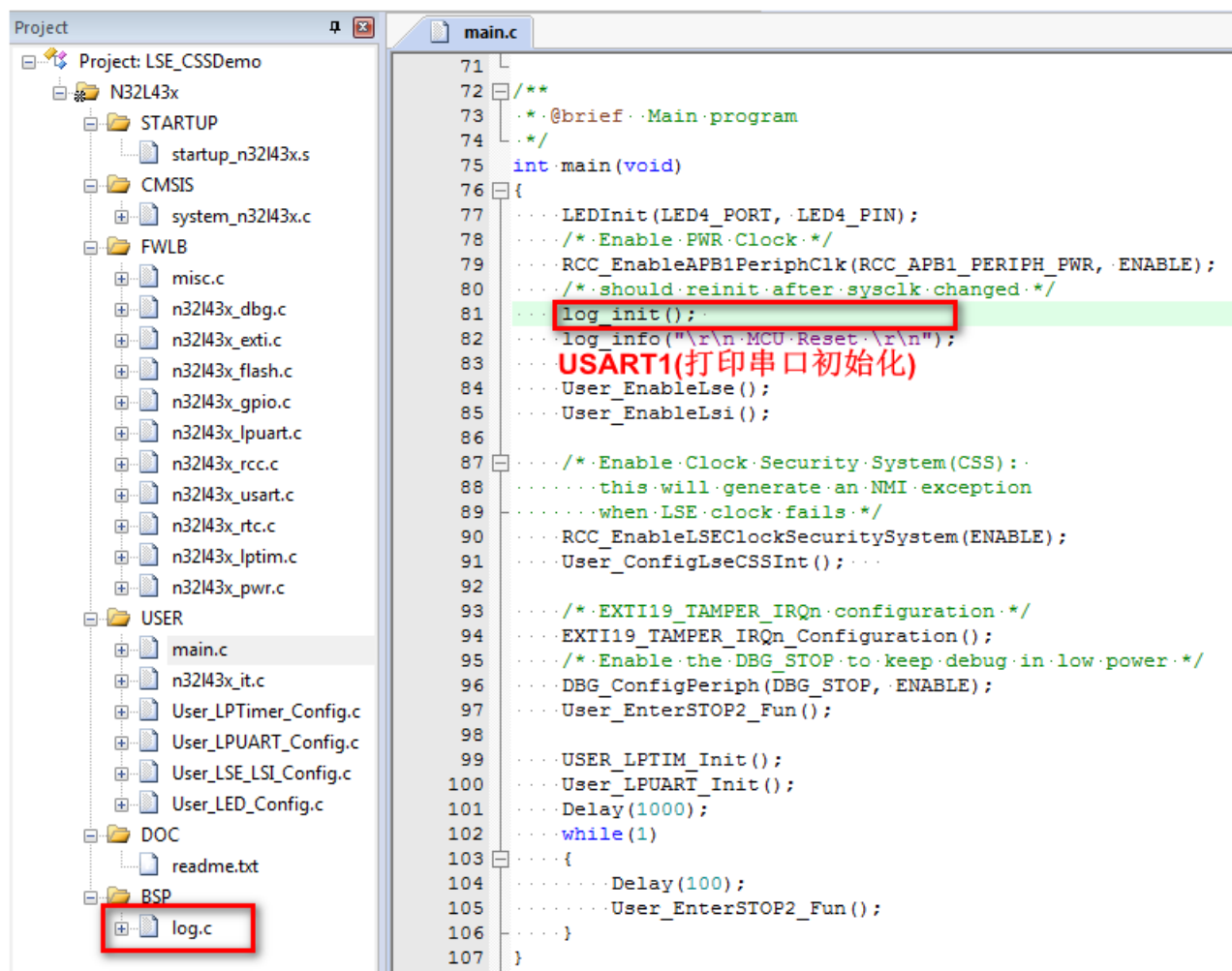
3 例程讲解

3.1 例程流程

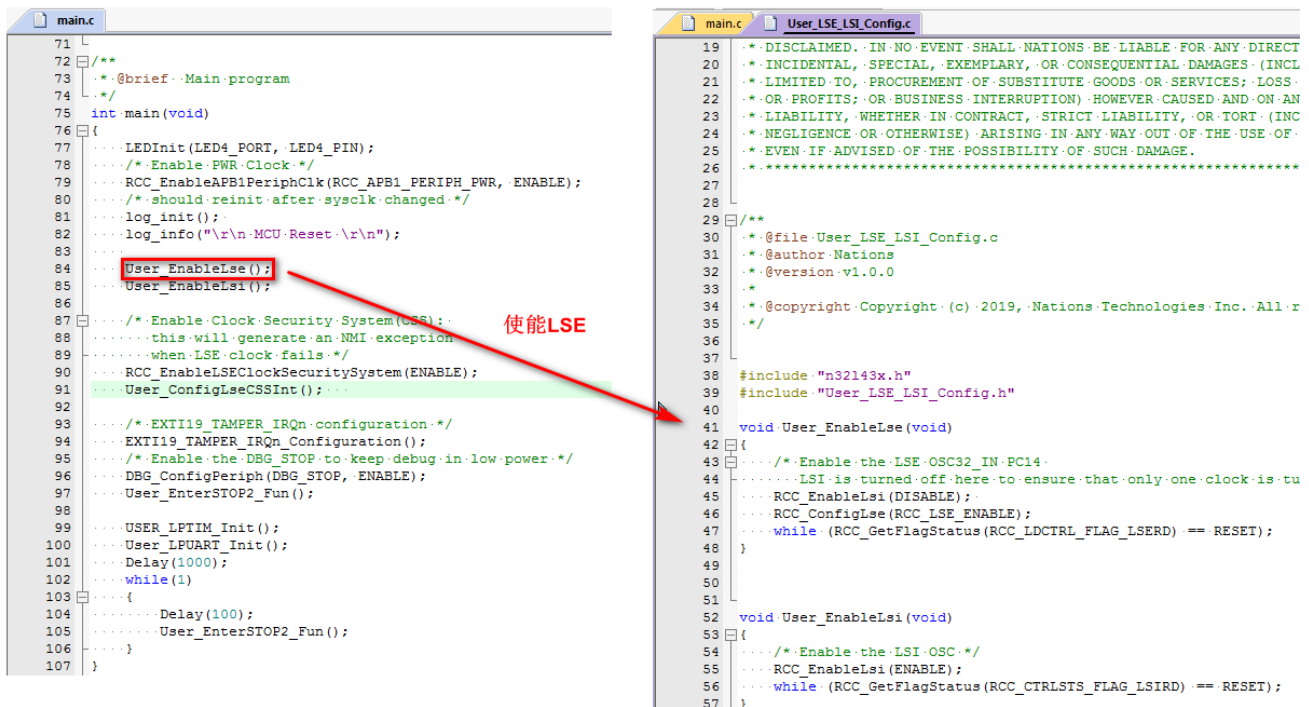


3.2 例程分析

3.2.1 USART1 日志串口初始化



3.2.2 使能 LSE

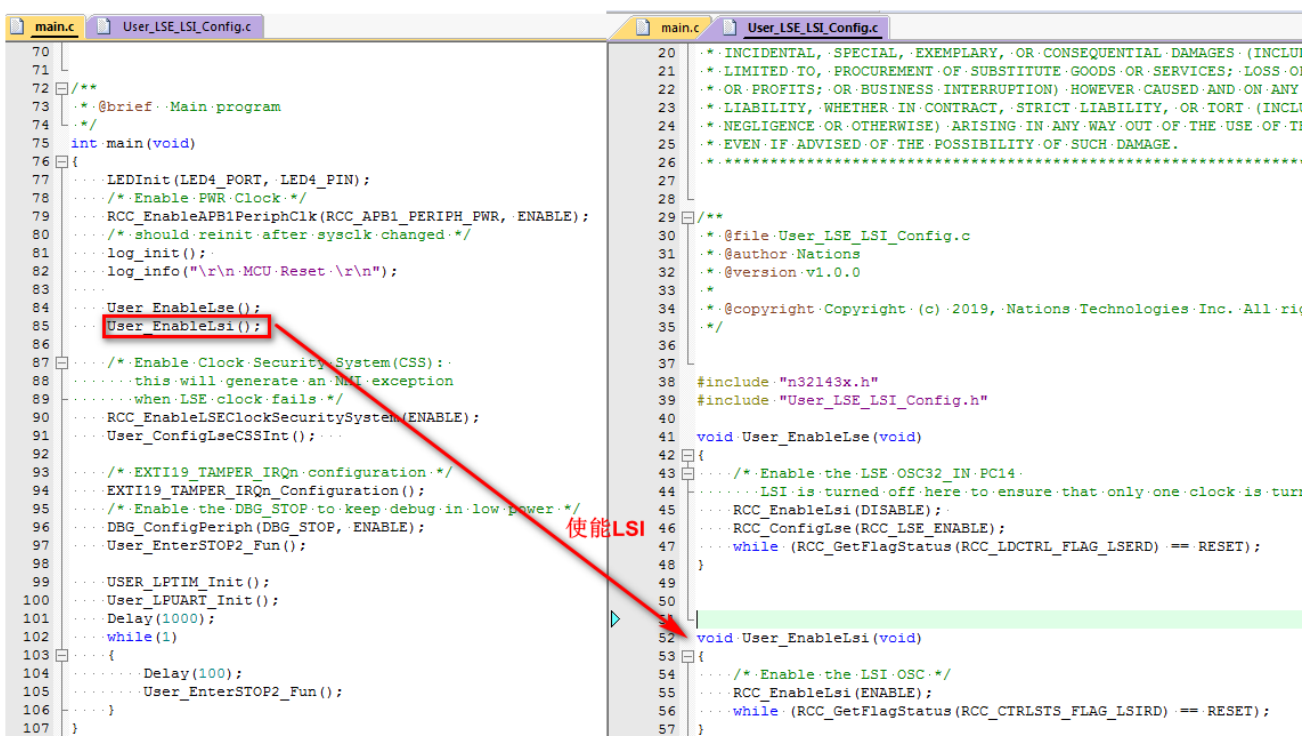


```

main.c
71
72 /**
73  * @brief Main program
74  */
75 int main(void)
76 {
77     LEDInit(LED4_PORT, LED4_PIN);
78     /* Enable PWR Clock */
79     RCC_EnableAPB1PeriphClk(RCC_APB1_PERIPH_PWR, ENABLE);
80     /* should reinit after sysclk changed */
81     log_init();
82     log_info("\r\n MCU Reset \r\n");
83
84     User_EnableLse();
85     User_EnableLsi();
86
87     /* Enable Clock Security System (CSS):
88      * this will generate an NMI exception
89      * when LSE clock fails */
90     RCC_EnableLSEClockSecuritySystem(ENABLE);
91     User_ConfigLseCSSInt();
92
93     /* EXTI19_TAMPER_IRQn configuration */
94     EXTI19_TAMPER_IRQn_Configuration();
95     /* Enable the DBG_STOP to keep debug in low power */
96     DBG_ConfigPeriph(DBG_STOP, ENABLE);
97     User_EnterSTOP2_Fun();
98
99     USER_LPTIM_Init();
100     User_LPUART_Init();
101     Delay(1000);
102     while(1)
103     {
104         Delay(100);
105         User_EnterSTOP2_Fun();
106     }
107 }

User_LSE_LSI_Config.c
19 /* DISCLAIMED. IN NO EVENT SHALL NATIONS BE LIABLE FOR ANY DIRECT
20 * INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCL
21 * LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS O
22 * OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON AN
23 * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INC
24 * NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF TI
25 * EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
26 *
27
28
29 /**
30  * @file User_LSE_LSI_Config.c
31  * @author Nations
32  * @version v1.0.0
33  *
34  * @copyright Copyright (c) 2019, Nations Technologies Inc. All r
35  */
36
37
38 #include "n32143x.h"
39 #include "User_LSE_LSI_Config.h"
40
41 void User_EnableLse(void)
42 {
43     /* Enable the LSE OSC32_IN_PC14
44      * LSI is turned off here to ensure that only one clock is tu
45      * RCC_EnableLsi(DISABLE);
46      * RCC_ConfigLse(RCC_LSE_ENABLE);
47      * while (RCC_GetFlagStatus(RCC_LDCTRL_FLAG_LSERD) == RESET);
48     }
49
50
51 void User_EnableLsi(void)
52 {
53     /* Enable the LSI OSC */
54     RCC_EnableLsi(ENABLE);
55     while (RCC_GetFlagStatus(RCC_CTRLSTS_FLAG_LSIRD) == RESET);
56 }
57
  
```

3.2.3 使能 LSI




```

main.c
70
71
72 /**
73  * @brief Main program
74  */
75 int main(void)
76 {
77     LEDInit(LED4_PORT, LED4_PIN);
78     /* Enable PWR Clock */
79     RCC_EnableAPB1PeriphClk(RCC_APB1_PERIPH_PWR, ENABLE);
80     /* should reinit after sysclk changed */
81     log_init();
82     log_info("\r\n MCU Reset \r\n");
83
84     User_EnableLse();
85     User_EnableLsi();
86
87     /* Enable Clock Security System (CSS):
88      * this will generate an NMI exception
89      * when LSE clock fails */
90     RCC_EnableLSEClockSecuritySystem(ENABLE);
91     User_ConfigLseCSSInt();
92
93     /* EXTI19_TAMPER_IRQn configuration */
94     EXTI19_TAMPER_IRQn_Configuration();
95     /* Enable the DBG_STOP to keep debug in low power */
96     DBG_ConfigPeriph(DBG_STOP, ENABLE);
97     User_EnterSTOP2_Fun();
98
99     USER_LPTIM_Init();
100     User_LPUART_Init();
101     Delay(1000);
102     while(1)
103     {
104         Delay(100);
105         User_EnterSTOP2_Fun();
106     }
107 }

User_LSE_LSI_Config.c
20 /* INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLU
21 * LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS O
22 * OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
23 * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCL
24 * NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF TI
25 * EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
26 *
27
28
29 /**
30  * @file User_LSE_LSI_Config.c
31  * @author Nations
32  * @version v1.0.0
33  *
34  * @copyright Copyright (c) 2019, Nations Technologies Inc. All ri
35  */
36
37
38 #include "n32143x.h"
39 #include "User_LSE_LSI_Config.h"
40
41 void User_EnableLse(void)
42 {
43     /* Enable the LSE OSC32_IN_PC14
44      * LSI is turned off here to ensure that only one clock is turn
45      * RCC_EnableLsi(DISABLE);
46      * RCC_ConfigLse(RCC_LSE_ENABLE);
47      * while (RCC_GetFlagStatus(RCC_LDCTRL_FLAG_LSERD) == RESET);
48     }
49
50
51 void User_EnableLsi(void)
52 {
53     /* Enable the LSI OSC */
54     RCC_EnableLsi(ENABLE);
55     while (RCC_GetFlagStatus(RCC_CTRLSTS_FLAG_LSIRD) == RESET);
56 }
57
  
```

3.2.4 使能 LSE-CSS 监控



```

70
71
72 /**
73  * @brief Main program
74  */
75 int main(void)
76 {
77     LEDInit(LED4_PORT, LED4_PIN);
78     /* Enable PWR Clock */
79     RCC_EnableAPB1PeriphClk(RCC_APB1_PERIPH_PWR, ENABLE);
80     /* Should reinit after sysclk changed */
81     log_init();
82     log_info("\r\n MCU Reset \r\n");
83
84     User_EnableLse();
85     User_EnableLsi();
86
87     /* Enable Clock Security System (CSS):
88      * this will generate an NMI exception
89      * when LSE clock fails */
90     RCC_EnableLSEClockSecuritySystem(ENABLE);
91     User_ConfigLseCSSInt();
92     /* EXTI19_TAMPER_IRQn configuration */
93     EXTI19_TAMPER_IRQn_Configuration();
94
95     /* Enable the DBG_STOP to keep debug in low power */
96     DBG_ConfigPeriph(DBG_STOP, ENABLE);
97     User_EnterSTOP2_Fun();
98
99     USER_LPTIM_Init();
100     User_LPUART_Init();
101     Delay(1000);
102     while(1)
103     {
104         Delay(100);
105         User_EnterSTOP2_Fun();
106     }
107 }

```

初始化LSE-CSS
并使能LSE-CSS中断

```

1613 /**
1614  * @brief Enables or disables the LSE Clock Security System.
1615  * @param Cmd new state of the LSE Clock Security System.
1616  * This parameter can be: ENABLE or DISABLE.
1617  */
1618 void RCC_EnableLSEClockSecuritySystem(FunctionalState Cmd)
1619 {
1620     /* Check the parameters */
1621     assert_param(IS_FUNCTIONAL_STATE(Cmd));
1622     *(__IO uint32_t*)LDCTRL_LSECLKSEN_BB = (uint32_t)Cmd;
1623 }

```

```

61 void User_ConfigLseCSSInt(void)
62 {
63     RCC->CLKINT |= (1 << 25);
64 }
65

```


3.2.5 进入 Stop2

```
main.c
65  /*Request to enter STOP2 mode*/
66  PWR_EnterSTOP2Mode(PWR_STOPENTRY_WFI, PWR_CTRL3_RAM1RET);
67  log_info("\n MCU Wakeup From STOP2 Mode \n");
68  }
69
70
71
72 /**
73  * @brief Main program
74  */
75 int main(void)
76 {
77  LEDInit(LED4_PORT, LED4_PIN);
78  /*Enable PWR Clock*/
79  RCC_EnableAPB1PeriphClk(RCC_APB1_PERIPH_PWR, ENABLE);
80  /*should reinit after sysclk changed*/
81  log_init();
82  log_info("\n MCU Reset \n");
83
84  User_EnableLse();
85  User_EnableLsi();
86
87  /*Enable Clock Security System(CSS):
88  *this will generate an NMI exception
89  *when LSE clock fails*/
90  RCC_EnableLSEClockSecuritySystem(ENABLE);
91  User_ConfigLseCSSInt();
92  /*EXTI19_TAMPER_IRQn configuration*/
93  EXTI19_TAMPER_IRQn_Configuration();
94
95  /*Enable the DBG_STOP to keep debug in low power*/
96  DBG_ConfigPeriph(DBG_STOP, ENABLE);
97  User_EnterSTOP2_Fun();
98
99  USER_LPTIM_Init();
100  User_LPUART_Init();
101  Delay(1000);
102  while(1)
103  {
104      Delay(100);
105      User_EnterSTOP2_Fun();
106  }
107 }
108
```

进入Stop2休眠模式

```
62 void User_EnterSTOP2_Fun(void)
63 {
64  log_info("\n MCU Goto STOP2 Mode \n");
65  /*Request to enter STOP2 mode*/
66  PWR_EnterSTOP2Mode(PWR_STOPENTRY_WFI, PWR_CTRL3_RAM1RET);
67  log_info("\n MCU Wakeup From STOP2 Mode \n");
68 }
```

3.2.6 初始化 LPTIM 和 LPUART

```

72 /**
73  * @brief Main program
74  */
75 int main(void)
76 {
77     LEDInit(LED4_PORT, LED4_PIN);
78     /* Enable FWR Clock */
79     RCC_EnableAPB1PeriphClk(RCC_APB1_PERIPH_FWR, ENABLE);
80     /* should reinit after sysclk changed */
81     log_init();
82     log_info("\r\n MCU Reset \r\n");
83     User_EnableLse();
84     User_EnableLsi();
85
86     /* Enable Clock Security System (CSS) :
87     * this will generate an NMI exception
88     * when LSE clock fails */
89     RCC_EnableLSEClockSecuritySystem(ENABLE);
90     User_ConfigLseCSSInt();
91     /* EXTI19_TAMPER_IRQn configuration */
92     EXTI19_TAMPER_IRQn_Configuration();
93
94     /* Enable the DBG_STOP to keep debug in low power */
95     DBG_ConfigPeriph(DBG_STOP, ENABLE);
96     User_EnterSTOP2_Fun();
97
98     USER_LPTIM_Init();
99     User_LPUART_Init();
100     Delay(1000);
101     while(1)
102     {
103         Delay(100);
104         User_EnterSTOP2_Fun();
105     }
106 }

```

初始化LPTIM
LPUART

```

68 void User_LPUART_Init(void)
69 {
70     LPUART_InitType LPUART_InitStructure;
71     /* Configure the GPIO ports */
72     GPIO_Configuration();
73
74     /* System Clocks Configuration */
75     RCC_Configuration(RCC_LPUARTCLK_SRC_LSE);
76
77     /* LPUART configuration */
78     LPUART_DeInit();
79     LPUART_StructInit(&LPUART_InitStructure);
80     LPUART_InitStructure.BaudRate = 9600;
81     LPUART_InitStructure.Parity = LPUART_PE_NO;
82     LPUART_InitStructure.RtsThreshold = LPUART_RTSTH_FIFOFU;
83     LPUART_InitStructure.HardwareFlowControl = LPUART_HFCTRL_NONE;
84     LPUART_InitStructure.Mode = LPUART_MODE_RX | LPUART_MODE_TX;
85     /* Configure LPUART */
86     LPUART_Init(&LPUART_InitStructure);
87 }

```

```

67 void USER_LPTIM_Init(void)
68 {
69     /* Enable interrupt */
70     LPTIMNVIC_Config(ENABLE);
71     RCC_ConfigLPTIMClk(RCC_LPTIMCLK_SRC_LSE);
72     RCC_EnableRETPeriphClk(RCC_RET_PERIPH_LPTIM, ENABLE);
73
74     LPTIM_SetPrescaler(LPTIM, LPTIM_PRESCALER_DIV4);
75     LPTIM_EnableIT_CMPM(LPTIM);
76     /* config lptim ARR and compare register */
77     LPTIM_Enable(LPTIM);
78     LPTIM_SetAutoReload(LPTIM, 65000);
79     LPTIM_SetCompare(LPTIM, 60000);
80     LPTIM_StartCounter(LPTIM, LPTIM_OPERATING_MODE_CONTINUOUS);
81 }

```

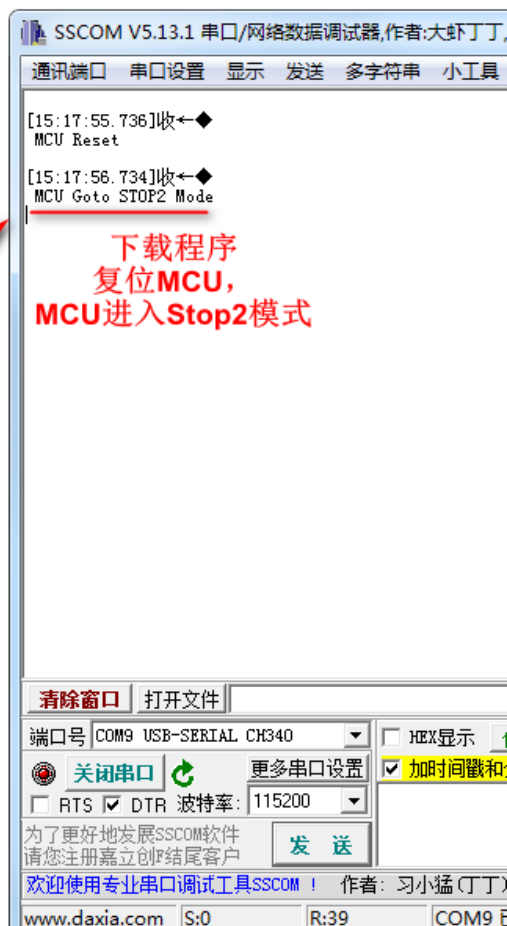
4 使用指南

4.1 复位 MCU

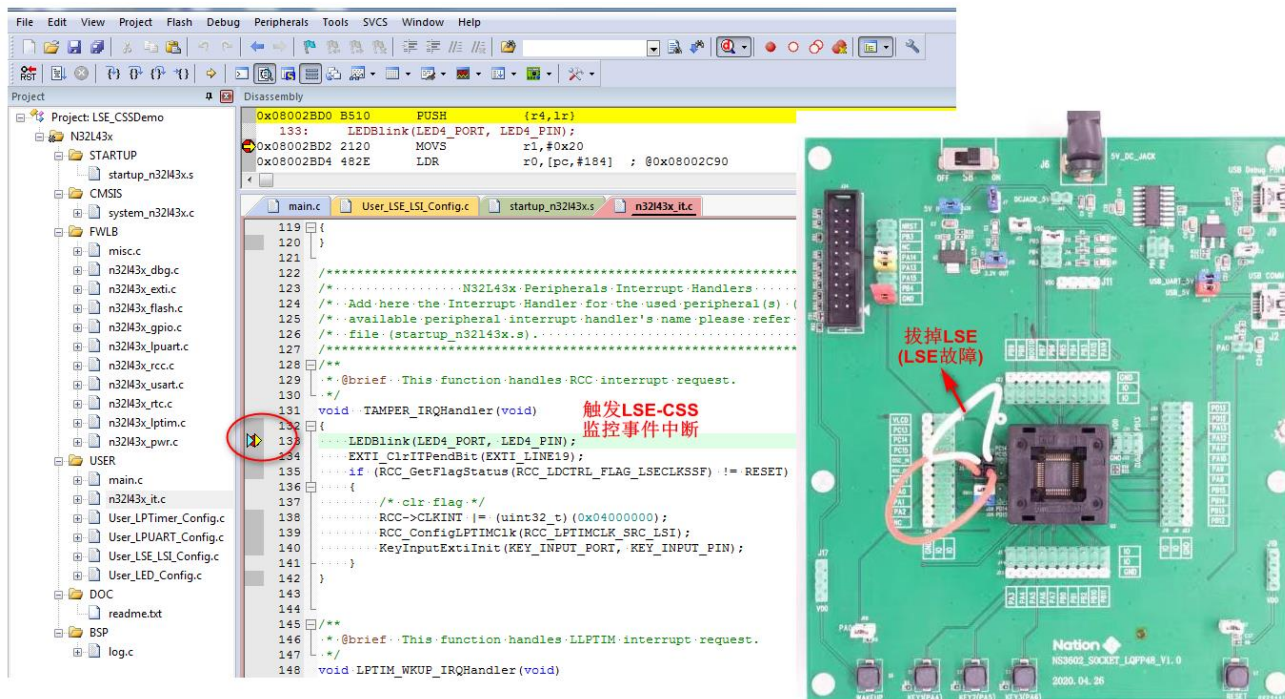
```

72 /**
73  * @brief Main program
74  */
75 int main(void)
76 {
77     LEDInit(LED4_PORT, LED4_PIN);
78     /* Enable PWR Clock */
79     RCC_EnableAPB1PeriphClk(RCC_APB1_PERIPH_PWR, ENABLE);
80     /* should reinit after sysclk changed */
81     log_init();
82     log_info("\r\n MCU Reset \r\n");
83
84     User_EnableLse();
85     User_EnableLsi();
86
87     /* Enable Clock Security System (CSS) :
88      * this will generate an NMI exception
89      * when LSE clock fails */
90     RCC_EnableLSEClockSecuritySystem(ENABLE);
91     User_ConfigLseCSSInt();
92     /* EXTI19_TAMPER_IRQn configuration */
93     EXTI19_TAMPER_IRQn_Configuration();
94
95     /* Enable the DBG_STOP to keep debug in low power */
96     DBG_ConfigPeriph(DBG_STOP, ENABLE);
97     User_EnterSTOP2_Fun();
98
99     USER_LPTIM_Init();
100    User_LPUART_Init();
101    Delay(1000);
102    while(1)
103    {
104        Delay(100);
105        User_EnterSTOP2_Fun();
106    }
107 }

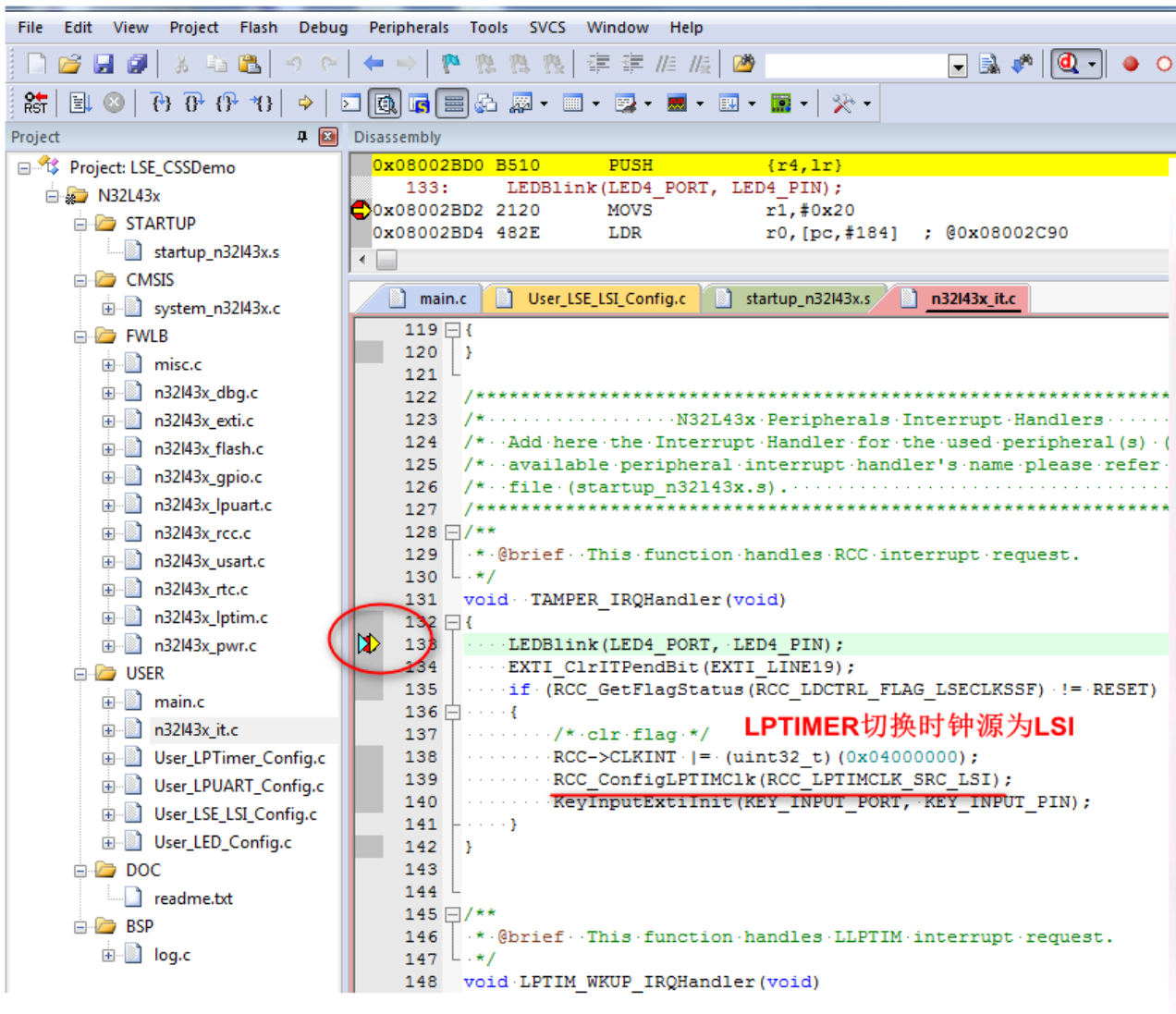
```



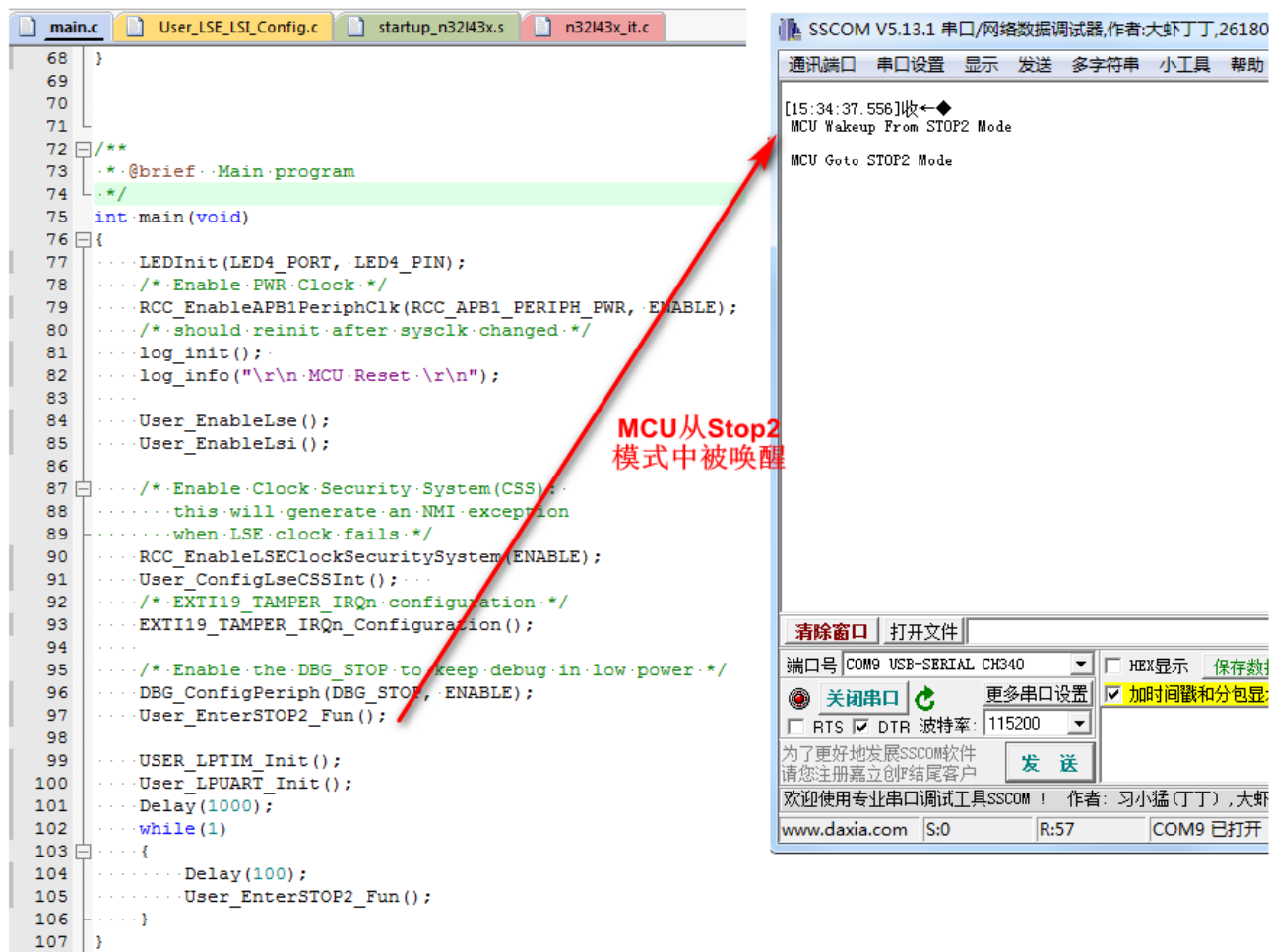
4.2 产生 LSE 故障



4.3 LPTIMER 切换时钟源



4.4 MCU 从 Stop2 中被唤醒



The image shows a development environment with a C code editor on the left and a serial terminal window on the right. A red arrow points from the code to the terminal output.

Code Editor (main.c):

```

68 }
69
70
71
72 /**
73  * @brief Main program
74  */
75 int main(void)
76 {
77     LEDInit(LED4_PORT, LED4_PIN);
78     /* Enable PWR Clock */
79     RCC_EnableAPB1PeriphClk(RCC_APB1_PERIPH_PWR, ENABLE);
80     /* should reinit after sysclk changed */
81     log_init();
82     log_info("\r\nMCU Reset\r\n");
83
84     User_EnableLse();
85     User_EnableLsi();
86
87     /* Enable Clock Security System (CSS):
88      * this will generate an NMI exception
89      * when LSE clock fails */
90     RCC_EnableLSEClockSecuritySystem(ENABLE);
91     User_ConfigLseCSSInt();
92     /* EXTI19_TAMPER_IRQn configuration */
93     EXTI19_TAMPER_IRQn_Configuration();
94
95     /* Enable the DBG_STOP to keep debug in low power */
96     DBG_ConfigPeriph(DBG_STOP, ENABLE);
97     User_EnterSTOP2_Fun();
98
99     USER_LPTIM_Init();
100    User_LPUART_Init();
101    Delay(1000);
102    while(1)
103    {
104        Delay(100);
105        User_EnterSTOP2_Fun();
106    }
107 }

```

Serial Terminal (SSCOM V5.13.1):

通讯端口: 串口设置 显示 发送 多字符串 小工具 帮助

[15:34:37.556]收←◆
MCU Wakeup From STOP2 Mode
MCU Goto STOP2 Mode

清除窗口 打开文件

端口号: COM9 USB-SERIAL CH340 ☐ HEX显示 保存数据
☒ 关闭串口 更多串口设置 ☒ 加时间戳和分包显示
☐ RTS ☒ DTR 波特率: 115200

为了更好地发展SSCOM软件
请您注册嘉立创结尾客户

发送

欢迎使用专业串口调试工具SSCOM! 作者: 习小猛(丁丁), 大虾
www.daxia.com S:0 R:57 COM9 已打开

MCU从Stop2模式中被唤醒

4.5 MCU 从进入 Stop2 休眠模式

The image shows a screenshot of a development environment. On the left, a C code file named `main.c` is open, showing the `main` function. The code includes various initialization steps and a `while(1)` loop. A red arrow points from the `User_EnterSTOP2_Fun();` line in the code to the serial terminal window on the right.

The serial terminal window, titled "SSCOM V5.13.1 串口/网络数据调试器, 作者: 大虾丁丁, 26180", displays the following messages:

```
[15:34:37.556]收←◆
MCU Wakeup From STOP2 Mode
MCU Goto STOP2 Mode
```

Below the terminal window, there are controls for the serial port, including a dropdown for the port (COM9), baud rate (115200), and checkboxes for RTS, DTR, and HEX display. A "发送" (Send) button is also visible.

MCU进入 Stop2 休眠模式

4.6 MCU 周期性唤醒

The image shows a code editor on the left and a serial terminal window on the right. The code editor displays the `main.c` file with the following key sections:

```

71
72 /**
73  * @brief Main program
74  */
75 int main(void)
76 {
77     LEDInit(LED4_PORT, LED4_PIN);
78     /* Enable PWR Clock */
79     RCC_EnableAPB1PeriphClk(RCC_APB1_PERIPH_PWR, ENABLE);
80     /* should reinit after sysclk changed */
81     log_init();
82     log_info("\r\n MCU Reset \r\n");
83
84     User_EnableLse();
85     User_EnableLsi();
86
87     /* Enable Clock Security System (CSS) :
88      * this will generate an NMI exception
89      * when LSE clock fails */
90     RCC_EnableLSEClockSecuritySystem(ENABLE);
91     User_ConfigLseCSSInt();
92     /* EXTI19 TAMPER IRQn configuration */
93     EXTI19_TAMPER_IRQn_Configuration();
94
95     /* Enable the DBG_STOP to keep debug in low power */
96     DBG_ConfigPeriph(DBG_STOP, ENABLE);
97     User_EnterSTOP2_Fun();
98
99     USER_LPTIM_Init();
100    User_LPUART_Init();
101    Delay(1000);
102    while(1)
103    {
104        Delay(100);
105        User_EnterSTOP2_Fun();
106    }
107 }

```

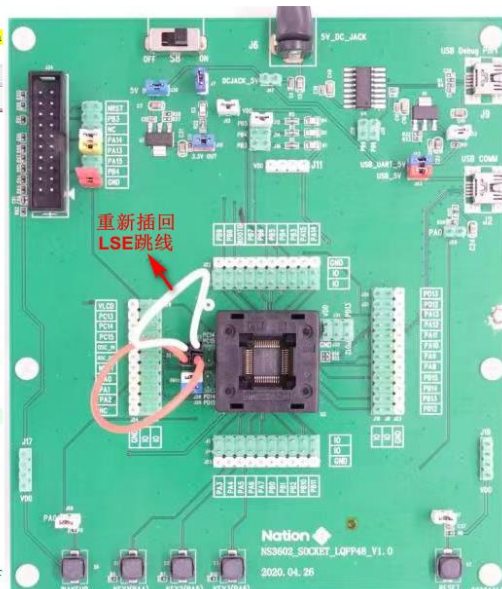
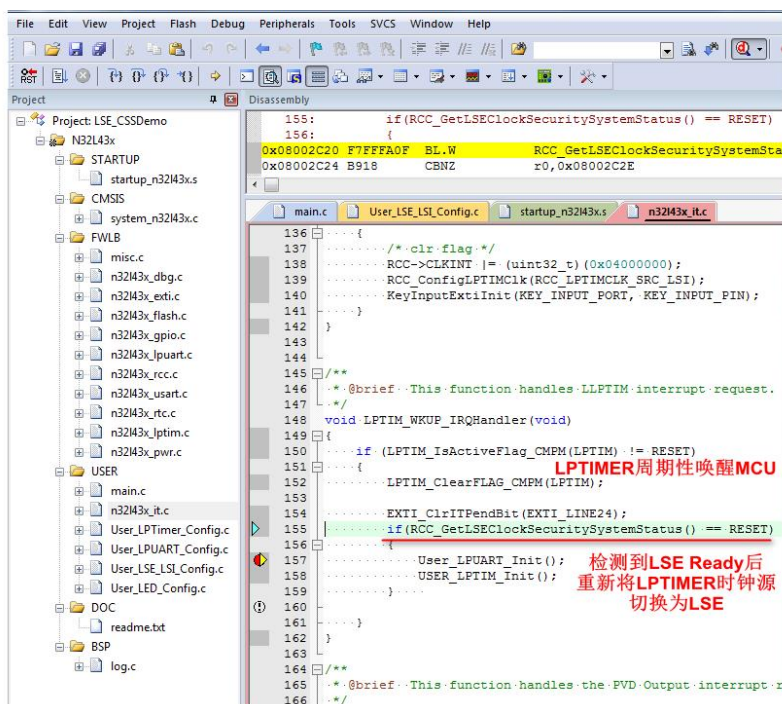
A red arrow points from the text "MCU被LPTIMER 周期性定时唤醒" to the `User_EnterSTOP2_Fun();` call in the `while(1)` loop.

The serial terminal window (SSCOM V5.13.1) shows the following sequence of events:

- [15:56:44.641]收 ← MCU Wakeup From STOP2 Mode
- MCU Goto STOP2 Mode
- [15:56:52.565]收 ← MCU Wakeup From STOP2 Mode
- MCU Goto STOP2 Mode
- [15:57:00.506]收 ← MCU Wakeup From STOP2 Mode
- MCU Goto STOP2 Mode
- [15:57:08.446]收 ← MCU Wakeup From STOP2 Mode
- MCU Goto STOP2 Mode
- [15:57:16.371]收 ← MCU Wakeup From STOP2 Mode
- MCU Goto STOP2 Mode
- [15:57:24.311]收 ← MCU Wakeup From STOP2 Mode
- MCU Goto STOP2 Mode
- [15:57:32.330]收 ← MCU Wakeup From STOP2 Mode
- MCU Goto STOP2 Mode
- [15:57:40.192]收 ← MCU Wakeup From STOP2 Mode
- MCU Goto STOP2 Mode

The terminal window also shows configuration settings: COM9, USB-SERIAL CH340, 115200 baud rate, and checkboxes for RTS, DTR, and HEX display.

4.7 LSE 恢复



4.8 LPTIMER 切换时钟源为 LSE

The screenshot displays a C code editor with the following code snippets:

```

65  ... /* Request to enter STOP2 mode */
66  ... PWR_EnterSTOP2Mode(PWR_STOPENTRY_WFI, PWR_CTRL3_RAM1RET);
67  ... log_info("\n MCU Wakeup From STOP2 Mode \n");
68  }
69
70
71
72  /**
73  * @brief Main program
74  */
75  int main(void)
76  {
77  ... LEDInit(LED4_PORT, LED4_PIN);
78  ... /* Enable PWR Clock */
79  ... RCC_EnableAPB1PeriphClk(RCC_APB1_PERIPH_PWR, ENABLE);
80  ... /* should reinit after sysclk changed */
81  ... log_init();
82  ... log_info("\r\n MCU Reset \r\n");
83  ...
84  ... User_EnableLse();
85  ... User_EnableLsi();
86
87  ... /* Enable Clock Security System (CSS) :
88  ... this will generate an NMI exception
89  ... when LSE clock fails */
90  ... RCC_EnableLSEClockSecuritySystem(ENABLE);
91  ... User_ConfigLseCSSInt();
92  ... /* EXTI19_TAMPER_IRQn configuration */
93  ... EXTI19_TAMPER_IRQn_Configuration();
94  ...
95  ... /* Enable the DBG_STOP to keep debug in low power */
96  ... DBG_ConfigPeriph(DBG_STOP, ENABLE);
97  ... User_EnterSTOP2_Fun();
98
99  ... USER_LPTIM_Init();
100  ... User_LPUART_Init();
101  ... Delay(1000);
102  ... while(1)
103  ... {
104  ... Delay(100);
105  ... User_EnterSTOP2_Fun();
106  ... }
107
108
109
110
111
112

```

The serial terminal window (SSCOM V5.13.1) shows the following log:

```

[16:07:43.072]收←◆
MCU Wakeup From STOP2 Mode

MCU Goto STOP2 Mode

[16:07:43.244]收←◆
MCU Wakeup From STOP2 Mode

MCU Goto STOP2 Mode

[16:07:51.184]收←◆
MCU Wakeup From STOP2 Mode

MCU Goto STOP2 Mode

```

A red arrow points from the code line `User_EnterSTOP2_Fun();` to the serial terminal window, indicating the execution of the function. A red text box with the text "LPTIMER时钟源切换为LSE后正常工作" (LPTIMER clock source switching to LSE works normally) is overlaid on the image.

5 历史版本

版本	日期	备注
V1.0	2020-11-30	创建文档

6 声 明

国民技术股份有限公司（下称“国民技术”）对此文档拥有专属产权。依据中华人民共和国的法律、条约以及世界其他法域相适用的管辖，此文档及其中描述的国民技术产品（下称“产品”）为公司所有。

国民技术在此并未授予专利权、著作权、商标权或其他任何知识产权许可。所提到或引用的第三方名称或品牌（如有）仅用作区别之目的。

国民技术保留随时变更、订正、增强、修改和改良此文档的权利，恕不另行通知。请使用人在下单购买前联系国民技术获取此文档的最新版本。

国民技术竭力提供准确可信的资讯，但即便如此，并不推定国民技术对此文档准确性和可靠性承担责任。

使用此文档信息以及生成产品时，使用者应当进行合理的设计、编程并测试其功能性和安全性，国民技术不对任何因使用此文档或本产品而产生的任何直接、间接、意外、特殊、惩罚性或衍生性损害结果承担责任。

国民技术对于产品在系统或设备中的应用效果没有任何故意或保证，如有任何应用在其发生操作不当或故障情况下，有可能致使人员伤亡、人身伤害或严重财产损失，则此类应用被视为“不安全使用”。

不安全使用包括但不限于：外科手术设备、原子能控制仪器、飞机或宇宙飞船仪器、所有类型的安全装置以及其他旨在支持或维持生命的应用。

所有不安全使用的风险应由使用人承担，同时使用人应使国民技术免于因为这类不安全使用而导致被诉、支付费用、发生损害或承担责任时的赔偿。

对于此文档和产品的任何明示、默示之保证，包括但不限于适销性、特定用途适用性和不侵权的保证责任，国民技术可在法律允许范围内进行免责。

未经明确许可，任何人不得以任何理由对此文档的全部或部分进行使用、复制、修改、抄录和传播。