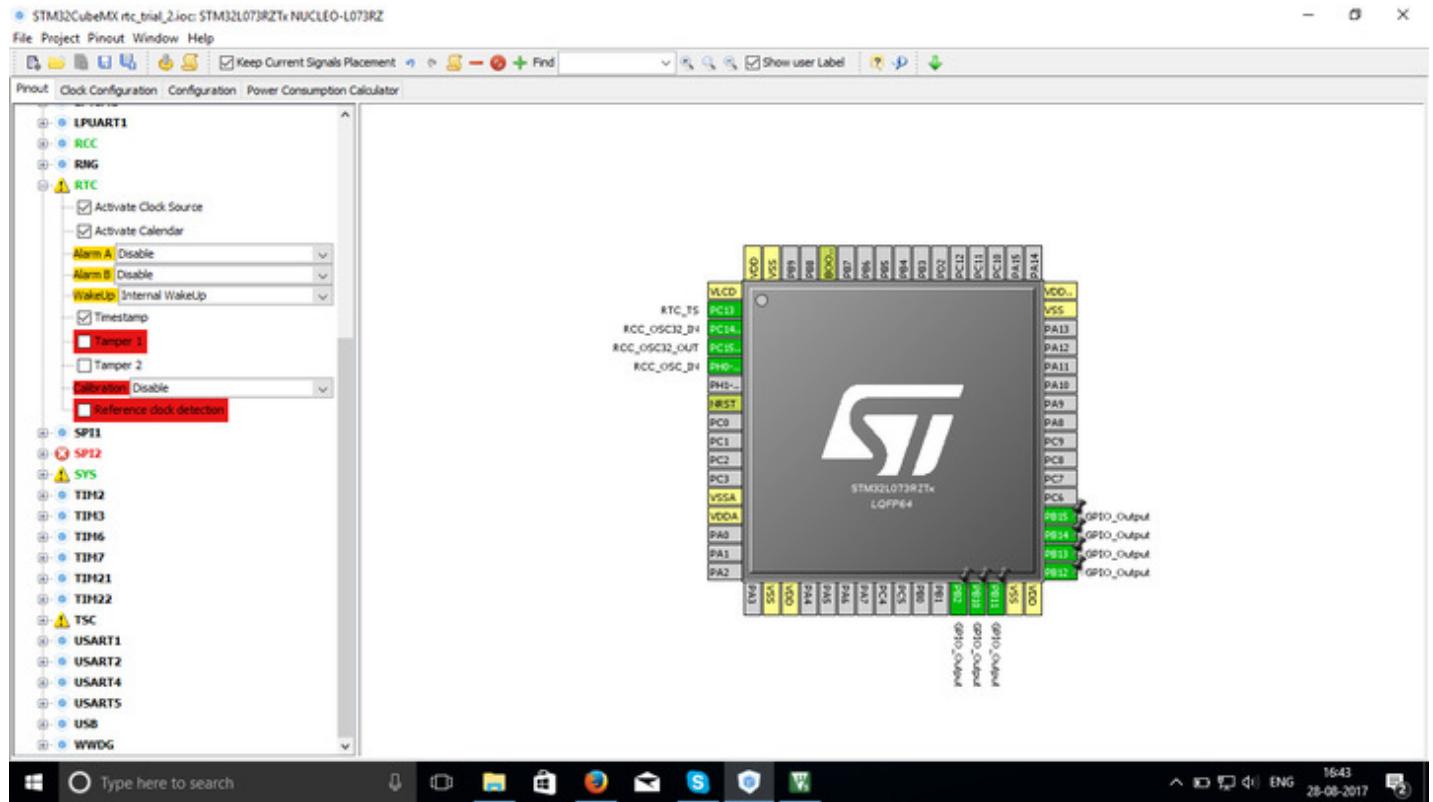
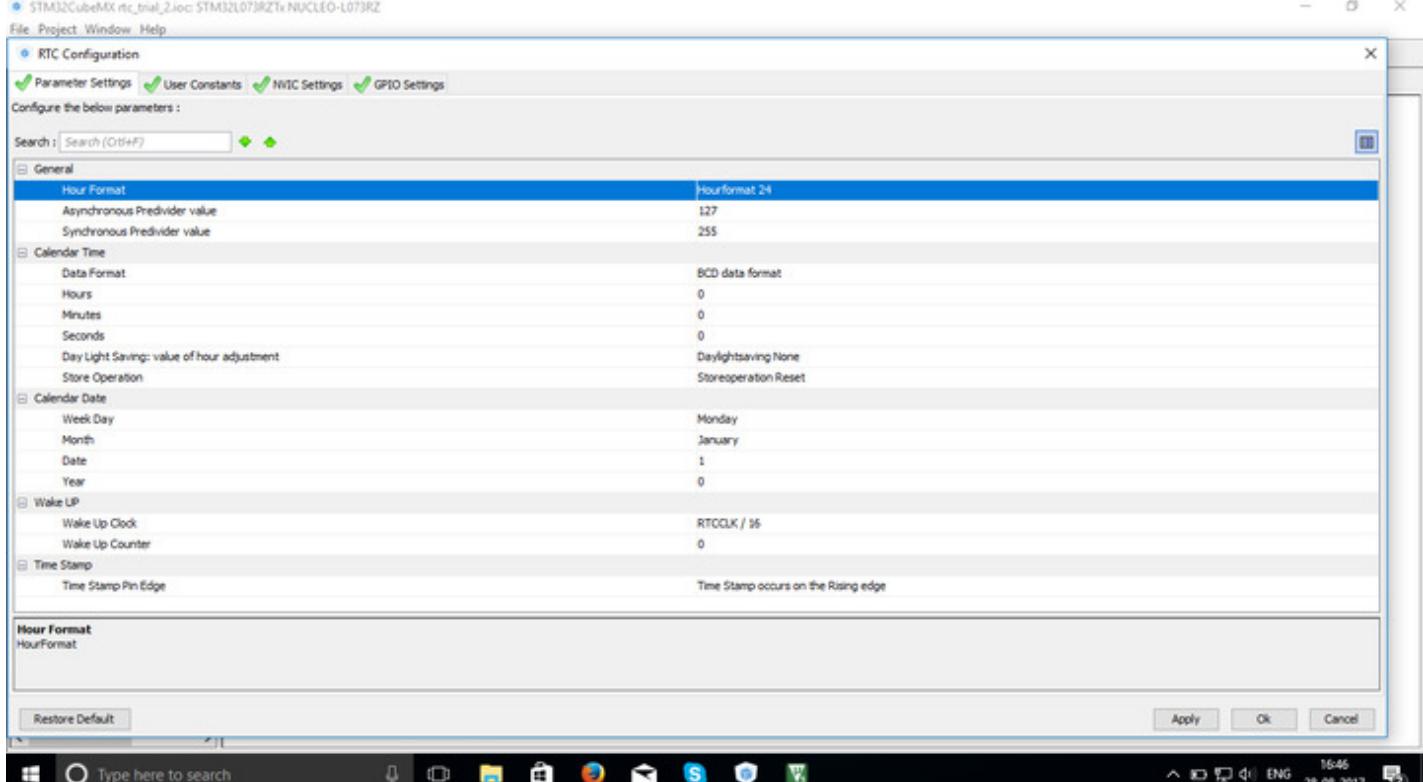
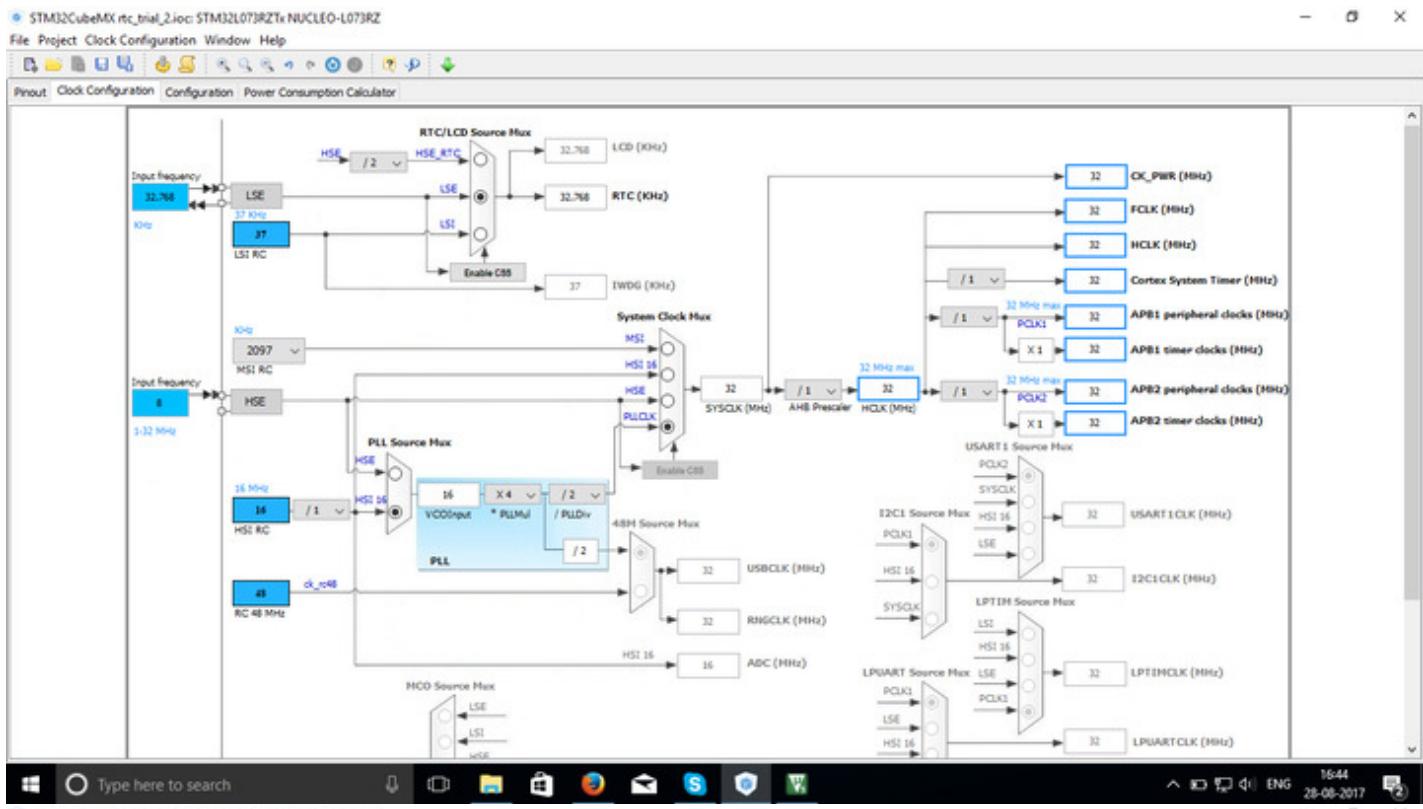


I am using stm32 nucleo-l073rz for a project. Currently I am working on internal rtc. I am facing some problems with this peripheral i am getting nothing on lcd but 00 as time filelds. I want to see a digital clock on lcd using internal rtc. I am using keil and stmcubemx with board. I have attached images with this post. Please have a look at it.





This is the main c file programme

```
/**  
*****  
* File Name      : main.c  
* Description    : Main program body  
*****  
*  
* COPYRIGHT(c) 2017 STMicroelectronics  
*  
* Redistribution and use in source and binary forms, with or without modification,  
* are permitted provided that the following conditions are met:  
*   1. Redistributions of source code must retain the above copyright notice,
```

\* this list of conditions and the following disclaimer.  
\* 2. Redistributions in binary form must reproduce the above copyright notice,  
\* this list of conditions and the following disclaimer in the documentation  
\* and/or other materials provided with the distribution.  
\* 3. Neither the name of STMicroelectronics nor the names of its contributors  
\* may be used to endorse or promote products derived from this software  
\* without specific prior written permission.  
\*  
\* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"  
\* AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE  
\* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE  
ARE  
\* DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE  
LIABLE  
\* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL  
\* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR  
\* SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER  
\* CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,  
\* OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE  
USE  
\* OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.  
\*  
\*\*\*\*\*  
\*/  
/\* Includes ----- \*/  
#include "main.h"  
#include "stm32l0xx\_hal.h"  
#include "lcd\_hd44780\_stm32l0.h"  
  
/\* USER CODE BEGIN Includes \*/  
  
/\* USER CODE END Includes \*/  
  
/\* Private variables ----- \*/  
RTC\_HandleTypeDef hrtc;  
  
/\* USER CODE BEGIN PV \*/  
/\* Private variables ----- \*/  
uint8\_t sec;  
/\* USER CODE END PV \*/  
  
/\* Private function prototypes ----- \*/  
void SystemClock\_Config(void);  
void Error\_Handler(void);  
static void MX\_GPIO\_Init(void);  
static void MX\_RTC\_Init(void);  
  
/\* USER CODE BEGIN PFP \*/  
/\* Private function prototypes ----- \*/  
  
/\* USER CODE END PFP \*/  
  
/\* USER CODE BEGIN 0 \*/  
  
/\* USER CODE END 0 \*/  
  
int main(void)  
{  
  
/\* USER CODE BEGIN 1 \*/

```
/* USER CODE END 1 */
```

```
/* MCU Configuration-----*/
```

```
/* Reset of all peripherals, Initializes the Flash interface and the Systick. */
HAL_Init();
```

```
/* Configure the system clock */
```

```
SystemClock_Config();
```

```
/* Initialize all configured peripherals */
```

```
MX_GPIO_Init();
```

```
MX_RTC_Init();
```

```
/* USER CODE BEGIN 2 */
```

```
RTC_TimeTypeDef RTC_Time;
```

```
HAL_RTC_GetTime(&hrtc,&RTC_Time,RTC_FORMAT_BCD);
```

```
sec=RTC_Time.Seconds;
```

```
LCDInit(0X01);
```

```
RTC_Time.Hours=0X05;
```

```
RTC_Time.Minutes=0x05;
```

```
RTC_Time.Seconds=0x00;
```

```
HAL_RTC_SetTime(&hrtc,&RTC_Time,RTC_FORMAT_BCD);
```

```
/* USER CODE END 2 */
```

```
/* Infinite loop */
```

```
/* USER CODE BEGIN WHILE */
```

```
while (1)
```

```
{
```

```
/* USER CODE END WHILE */
```

```
/* USER CODE BEGIN 3 */
```

```
HAL_RTC_GetTime(&hrtc,&RTC_Time,RTC_FORMAT_BCD);
```

```
sec=RTC_Time.Seconds;
```

```
LCDGotoXY(5,1);
```

```
LCDWriteInt(sec,2);
```

```
HAL_Delay(500);
```

```
LCDCmd(0X01);
```

```
}
```

```
/* USER CODE END 3 */
```

```
}
```

```
/** System Clock Configuration
```

```
*/
```

```
void SystemClock_Config(void)
```

```
{
```

```
RCC_OscInitTypeDef RCC_OscInitStruct;
```

```
RCC_ClkInitTypeDef RCC_ClkInitStruct;
```

```
RCC_PeriphCLKInitTypeDef PeriphClkInit;
```

```
/**Configure the main internal regulator output voltage
```

```
*/
```

```
_HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1);
```

```
/**Initializes the CPU, AHB and APB busses clocks
```

```
*/
```

```
RCC_OscInitStruct.OscillatorType = RCC OSCILLATORTYPE_HSI|RCC OSCILLATORTYPE_LSE;
RCC_OscInitStruct.LSEState = RCC_LSE_ON;
RCC_OscInitStruct.HSIStruct = RCC_HSI_ON;
RCC_OscInitStruct.HSICalibrationValue = 16;
RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSI;
RCC_OscInitStruct.PLL.PLLMUL = RCC_PLLMUL_4;
RCC_OscInitStruct.PLL.PLLDIV = RCC_PLLDIV_2;
if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
{
    Error_Handler();
}

/**Initializes the CPU, AHB and APB busses clocks
*/
RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
    |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;
RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;

if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_1) != HAL_OK)
{
    Error_Handler();
}

PeriphClkInit.PeriphClockSelection = RCC_PERIPHCLK_RTC;
PeriphClkInit.RTCClockSelection = RCC_RTCCLKSOURCE_LSE;
if (HAL_RCCEx_PeriphCLKConfig(&PeriphClkInit) != HAL_OK)
{
    Error_Handler();
}

/**Configure LSE Drive Capability
*/
__HAL_RCC_LSEDRIVE_CONFIG(RCC_LSEDRIVE_LOW);

/**Configure the Systick interrupt time
*/
HAL_SYSTICK_Config(HAL_RCC_GetHCLKFreq()/1000);

/**Configure the Systick
*/
HAL_SYSTICK_CLKSourceConfig(SYSTICK_CLKSOURCE_HCLK);

/* SysTick_IRQn interrupt configuration */
HAL_NVIC_SetPriority(SysTick_IRQn, 0, 0);
}

/* RTC init function */
static void MX_RTC_Init(void)
{
    RTC_TimeTypeDef sTime;
    RTC_DateTypeDef sDate;

    /**Initialize RTC Only
    */
    hrtc.Instance = RTC;
```

```
hrtc.Init.HourFormat = RTC_HOURFORMAT_24;
hrtc.Init.AsynchPrediv = 127;
hrtc.Init.SynchPrediv = 255;
hrtc.Init.OutPut = RTC_OUTPUT_DISABLE;
hrtc.Init.OutPutRemap = RTC_OUTPUT_REMAP_NONE;
hrtc.Init.OutPutPolarity = RTC_OUTPUT_POLARITY_HIGH;
hrtc.Init.OutPutType = RTC_OUTPUT_TYPE_OPENDRAIN;
if (HAL_RTC_Init(&hrtc) != HAL_OK)
{
    Error_Handler();
}

/**Initialize RTC and set the Time and Date
 */
sTime.Hours = 0x0;
sTime.Minutes = 0x0;
sTime.Seconds = 0x0;
sTime.DayLightSaving = RTC_DAYLIGHTSAVING_NONE;
sTime.StoreOperation = RTC_STOREOPERATION_RESET;
if (HAL_RTC_SetTime(&hrtc, &sTime, RTC_FORMAT_BCD) != HAL_OK)
{
    Error_Handler();
}

sDate.WeekDay = RTC_WEEKDAY_MONDAY;
sDate.Month = RTC_MONTH_JANUARY;
sDate.Date = 0x1;
sDate.Year = 0x0;

if (HAL_RTC_SetDate(&hrtc, &sDate, RTC_FORMAT_BCD) != HAL_OK)
{
    Error_Handler();
}

/**Enable the WakeUp
 */
if (HAL_RTCEx_SetWakeUpTimer(&hrtc, 0, RTC_WAKEUPCLOCK_RTCCLK_DIV16) != HAL_OK)
{
    Error_Handler();
}

/**Enable theTimeStamp
 */
if (HAL_RTCEx_SetTimeStamp(&hrtc, RTC_TIMESTAMPEDGE_RISING,
RTC_TIMESTAMPPIN_DEFAULT) != HAL_OK)
{
    Error_Handler();
}

/** Configure pins as
 * Analog
 * Input
 * Output
 * EVENT_OUT
 * EXTI
*/

```

```

static void MX_GPIO_Init(void)
{
    GPIO_InitTypeDef GPIO_InitStruct;

    /* GPIO Ports Clock Enable */
    __HAL_RCC_GPIOC_CLK_ENABLE();
    __HAL_RCC_GPIOH_CLK_ENABLE();
    __HAL_RCC_GPIOB_CLK_ENABLE();

    /*Configure GPIO pin Output Level*/
    HAL_GPIO_WritePin(GPIOB, GPIO_PIN_2|GPIO_PIN_10|GPIO_PIN_11|GPIO_PIN_12
                      |GPIO_PIN_13|GPIO_PIN_14|GPIO_PIN_15, GPIO_PIN_RESET);

    /*Configure GPIO pins : PB2 PB10 PB11 PB12
                           PB13 PB14 PB15 */
    GPIO_InitStruct.Pin = GPIO_PIN_2|GPIO_PIN_10|GPIO_PIN_11|GPIO_PIN_12
                      |GPIO_PIN_13|GPIO_PIN_14|GPIO_PIN_15;
    GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
    GPIO_InitStruct.Pull = GPIO_NOPULL;
    GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
    HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);

}

/* USER CODE BEGIN 4 */

/* USER CODE END 4 */

/**
 * @brief This function is executed in case of error occurrence.
 * @param None
 * @retval None
 */
void Error_Handler(void)
{
    /* USER CODE BEGIN Error_Handler */
    /* User can add his own implementation to report the HAL error return state */
    while(1)
    {
    }
    /* USER CODE END Error_Handler */
}

#ifndef USE_FULL_ASSERT

/**
 * @brief Reports the name of the source file and the source line number
 * where the assert_param error has occurred.
 * @param file: pointer to the source file name
 * @param line: assert_param error line source number
 * @retval None
 */
void assert_failed(uint8_t* file, uint32_t line)
{
    /* USER CODE BEGIN 6 */
    /* User can add his own implementation to report the file name and line number,
     ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */
    /* USER CODE END 6 */
}

```

{

#endif

/\*\*

\* @}

\*/

/\*\*

\* @}

\*/

\*\*\*\*\* (C) COPYRIGHT STMicroelectronics \*\*\*\*\*END OF FILE\*\*\*\*/