

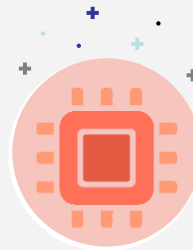


In The Name of God

Faculty of Electrical Engineering
Microprocessor Laboratory



V1 (@2025)



Microprocessor LAB

Lecture 7:

Internal RTC in STM32

By: M.Razeghizadeh

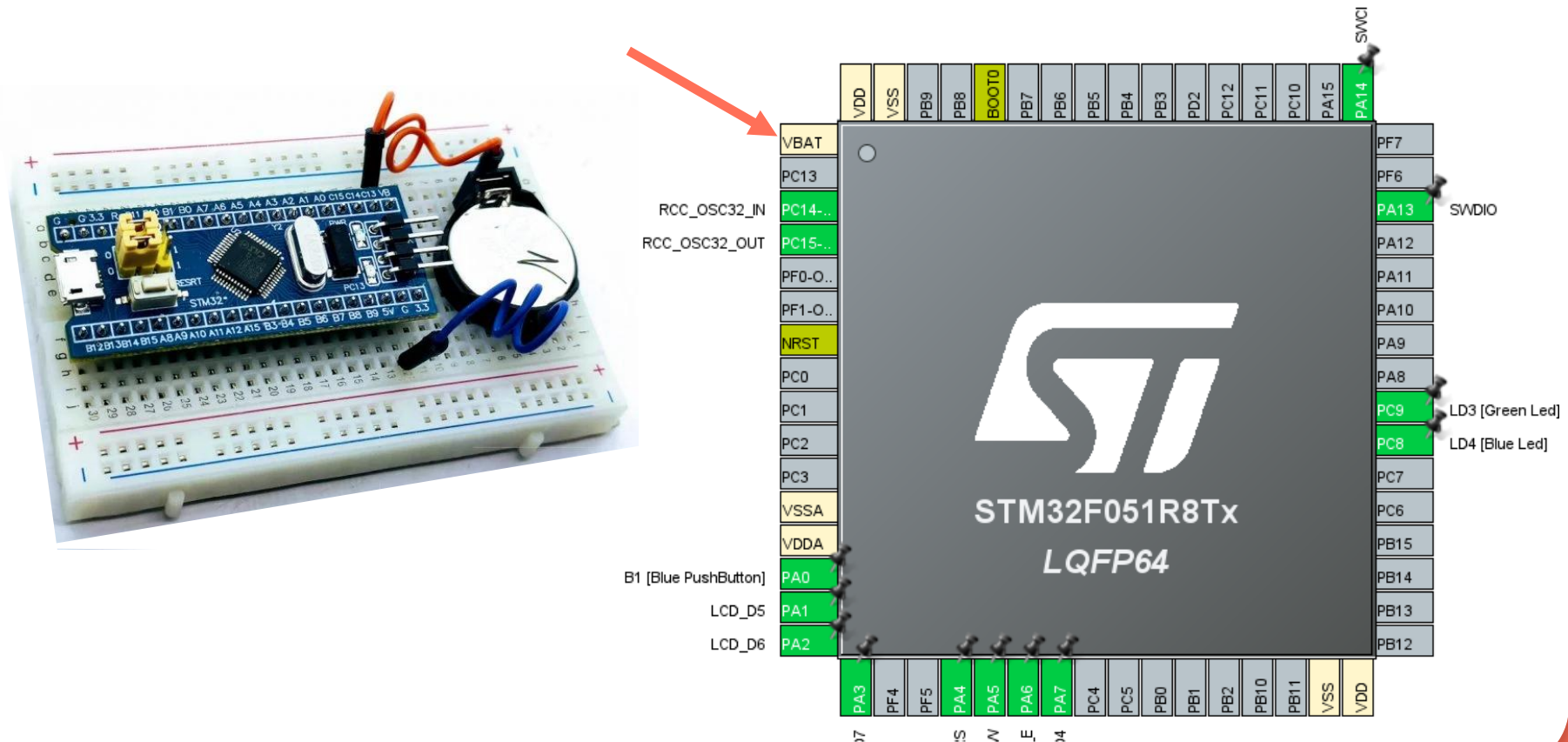
Start now!



نحوه‌ی استفاده از RTC داخلی در
STM32 همراه با آلارم
و نمایش روی LCD

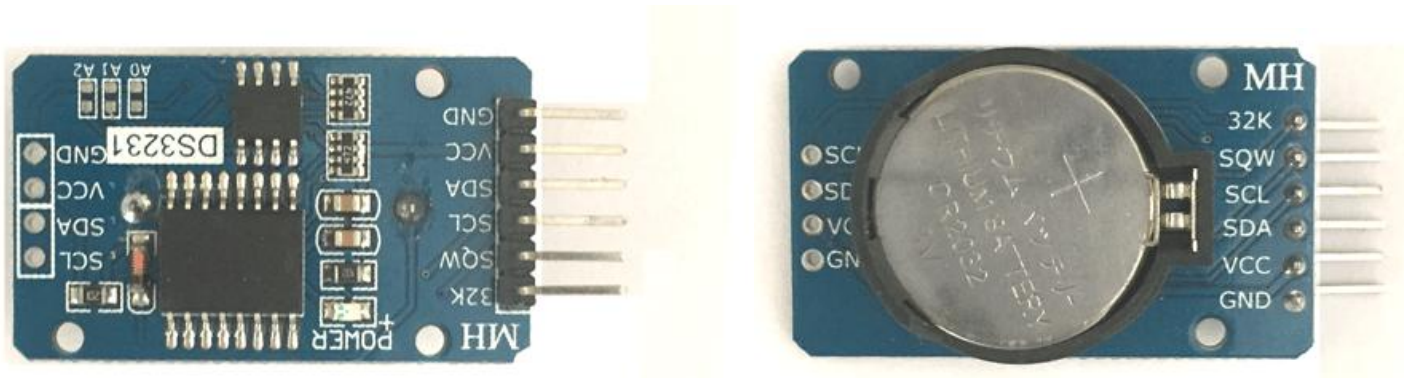
RTC چیست؟ (Real-Time Clock)

- واحد RTC در میکروکنترلرهای STM32 یک محیط سخت‌افزاری اختصاصی برای نگهداری زمان و تاریخ است؛ حتی زمانی که تغذیه اصلی قطع شود. این واحد با کمک منبع تغذیه پشتیبان VBAT می‌تواند اطلاعات زمان را برای مدت طولانی حفظ کند.



مزیت وجود واحد RTC داخلی در میکروکنترلرهای STM32

- این ویژگی باعث می‌شود ساعت داخلی بدون نیاز به کالیبراسیون نرم‌افزاری یا ماژول جانبی، پایدار و قابل‌اعتماد باقی بماند.
- در مقابل، بسیاری از میکروکنترلرهای متداول مانند AVR، PIC و برخی میکروهای ARM سطح پایین، واحد RTC مستقل یا منبع تغذیه پشتیبان داخلی ندارند. در این موارد برای داشتن زمان واقعی دقیق، معمولاً باید از تراشه‌ها و ماژول‌های خارجی مانند DS1307 یا DS3231 استفاده کرد؛ که این موضوع موجب افزایش هزینه، پیچیدگی طراحی، و اشغال خطوط ارتباطی (I2C) می‌شود.



واحد RTC در STM32

➤ در این آموزش، با کمک STM32CubeMX و توابع HAL مراحل زیر را به صورت قدم به قدم انجام می‌دهیم:



- راه‌اندازی و پیکربندی RTC داخلی
- انتخاب منبع کلاک مناسب (LSE یا LSI)
- برنامه‌ریزی و فعال‌سازی آلارم‌های RTC
- استفاده از باتری پشتیبان برای حفظ زمان
- نمایش ساعت و تاریخ روی LCD

واحد RTC در STM32

➤ RTC یک واحد سخت‌افزاری ویژه در میکروکنترلر است که وظیفه نگه‌داری زمان و تاریخ را بر عهده دارد. این واحد می‌تواند موارد زیر را دنبال کند:

✓ ثانیه، دقیقه و ساعت

✓ روز، ماه و سال

➤ ویژگی‌های اصلی RTC:

✓ برخلاف تایمرهای نرم‌افزاری، به صورت پیوسته و مستقل کار می‌کند.

✓ مصرف توان بسیار پایینی دارد.

✓ حتی زمانی که پردازنده خاموش باشد، فعال باقی می‌ماند.

✓ با یک باتری کوچک سکه‌ای (VBAT) می‌تواند ساعت و تاریخ را پس از ریست یا قطع برق نیز حفظ کند.

✓ قابلیت تولید آلارم و بیدارباش دوره‌ای را دارد.

➤ **کاربرد:** RTC یک گزینه ایده‌آل برای سیستم‌های کم‌مصرف و دستگاه‌هایی است که باید زمان را در طول خاموشی حفظ کنند.

انتخاب منبع کلاک RTC (LSE یا LSI)

برای استفاده از RTC داخلی STM32، باید یک منبع کلاک انتخاب کنید. RTC می‌تواند از LSE (کریستال خارجی کم‌سرعت) یا LSI (اسیلاتور داخلی RC کم‌سرعت) تغذیه شود. انتخاب منبع مناسب بسته به نیاز پروژه، به دقت، مصرف انرژی و هزینه دارد.

➤ LSE کریستال خارجی کم‌سرعت

- کار با فرکانس ۳۲/۷۶۸ کیلوهرتز و دقت بسیار بالا
- بهترین گزینه برای کاربردهای زمان واقعی که نیاز به دقت بالا دارند
- نیاز به کریستال خارجی متصل به برد STM32 دارد
- هزینه و فضای برد کمی بالاتر، اما دقت و پایداری طولانی‌مدت را تضمین می‌کند

➤ LSI اسیلاتور داخلی کم‌سرعت

- اسیلاتور RC داخلی، بدون نیاز به سخت‌افزار اضافی
- دقت کمتر نسبت به LSE (ممکن است با گذر زمان انحراف داشته باشد)
- مناسب برای زمان‌بندی‌های ساده یا پروژه‌های کم‌هزینه
- همیشه در STM32 موجود است و برای بردهای با فضای محدود مفید است

معیار انتخاب منبع کلاک

- انتخاب بین LSE و LSI به نیازهای پروژه بستگی دارد:

- **انتخاب LSE:**

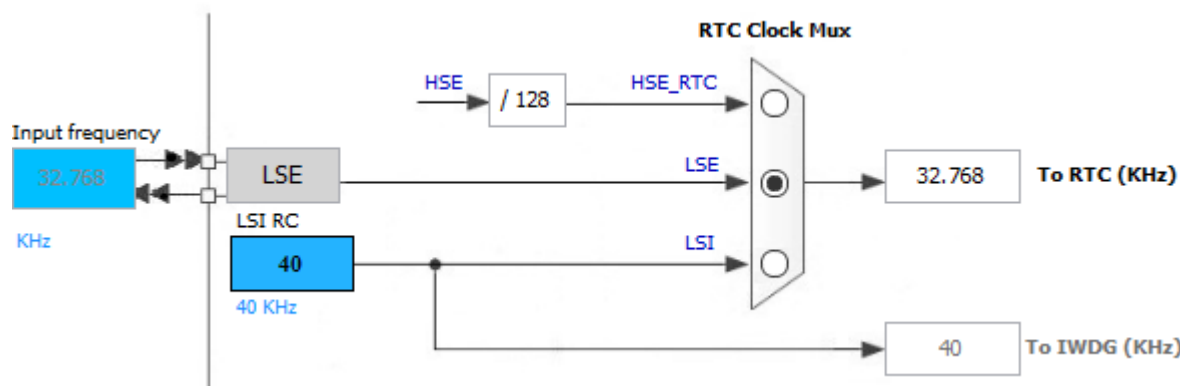
- ✓ اگر دقت زمان واقعی حیاتی است. (مثلاً ساعت دقیق، ثبت رویدادهای حساس به زمان)
- ✓ اگر هدف کارکرد طولانی مدت و پایدار باشد.
- ✓ وقتی فضای برد و هزینه اضافه برای کریستال خارجی مشکلی ایجاد نمی کند.

- **انتخاب LSI:**

- ✓ اگر دقت خیلی بالا مهم نیست و کمی خطا یا drift قابل قبول است.
- ✓ برای پروژه های کم هزینه یا نمونه اولیه.
- ✓ وقتی فضای برد محدود است و نمی خواهید قطعات خارجی اضافه کنید.

- به زبان ساده: برای دقت بالا و پروژه های جدی LSE، برای سادگی و صرفه جویی LSI.

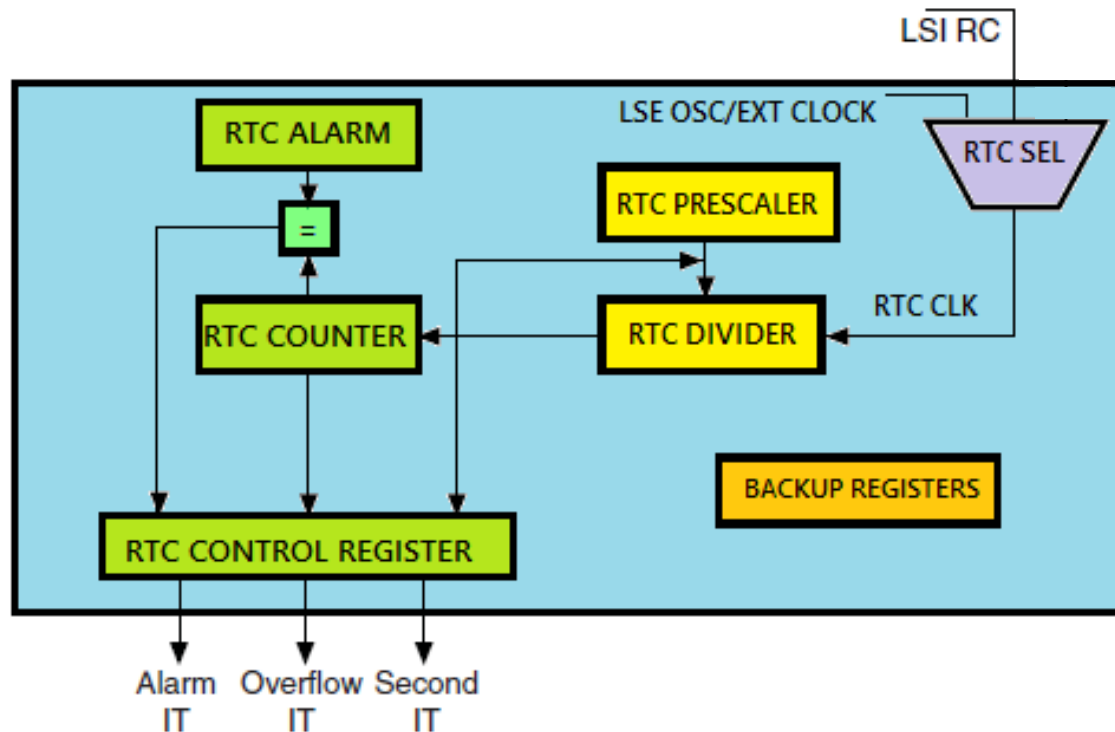
معيار انتخاب منبع کلاک



ویژگی	LSE (کریستال خارجی کم سرعت)	LSI (اسیلاتور داخلی کم سرعت)
فرکانس	۳۲,۷۶۸ کیلوهرتز (کریستال استاندارد ساعت)	تقریباً ۴۰ کیلوهرتز (RC داخلی)
دقت	بسیار بالا و پایدار در طول زمان	پایین تر، ممکن است با دما و ولتاژ دچار انحراف شود
سخت افزار مورد نیاز	کریستال خارجی ۳۲,۷۶۸ کیلوهرتز	بدون نیاز به سخت افزار خارجی
مصرف انرژی	کم	کم
هزینه	کمی بالاتر (به دلیل کریستال اضافی)	بدون هزینه اضافه
مناسب برای	ساعت های دقیق، ثبت داده، اینترنت اشیا (IoT)	آلارم های ساده، پروژه های کم هزینه

شرح عملکرد واحد RTC

این نمودار ساختار کلی واحد RTC را در STM32 نشان می‌دهد. RTC یک واحد مستقل است که با یک کلاک کم‌سرعت کار می‌کند و حتی در زمان خاموش بودن MCU نیز فعال می‌ماند.



شرح عملکرد واحد RTC

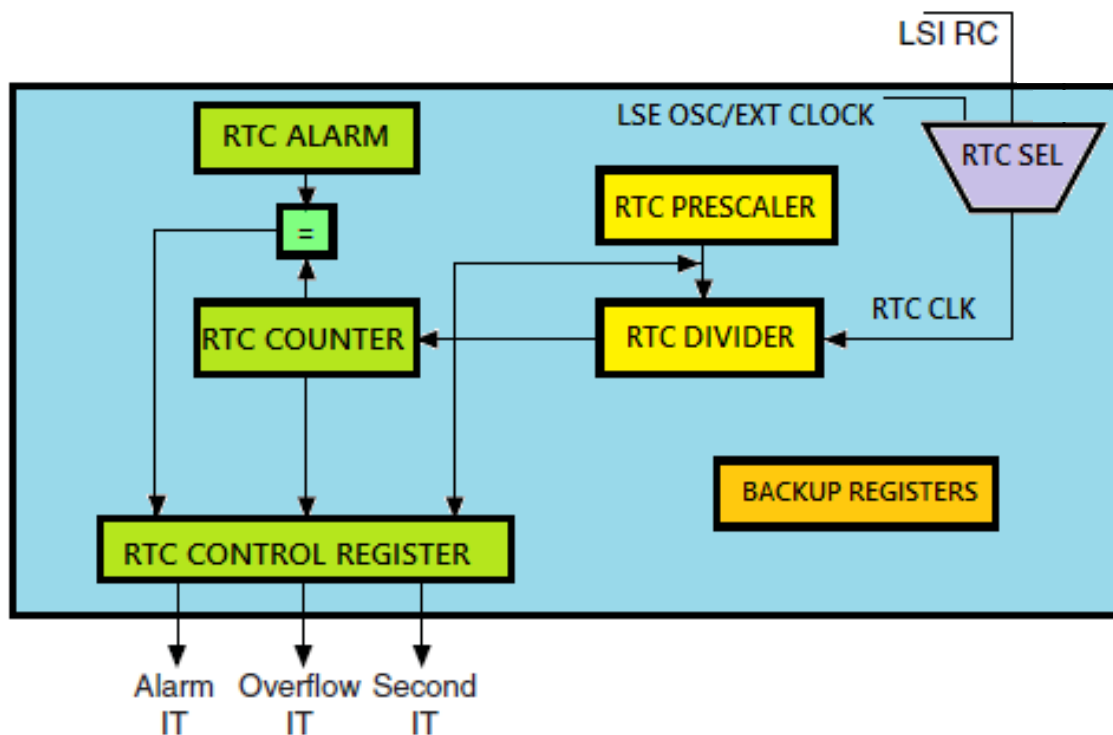
۱. انتخاب منبع کلاک (RTC SEL)

RTC می‌تواند از یکی از دو منبع کلاک زیر استفاده کند:

✓ LSE: کریستال خارجی ۳۲/۷۶۸ kHz

✓ LSI: اسیلاتور داخلی حدود ۴۰ kHz

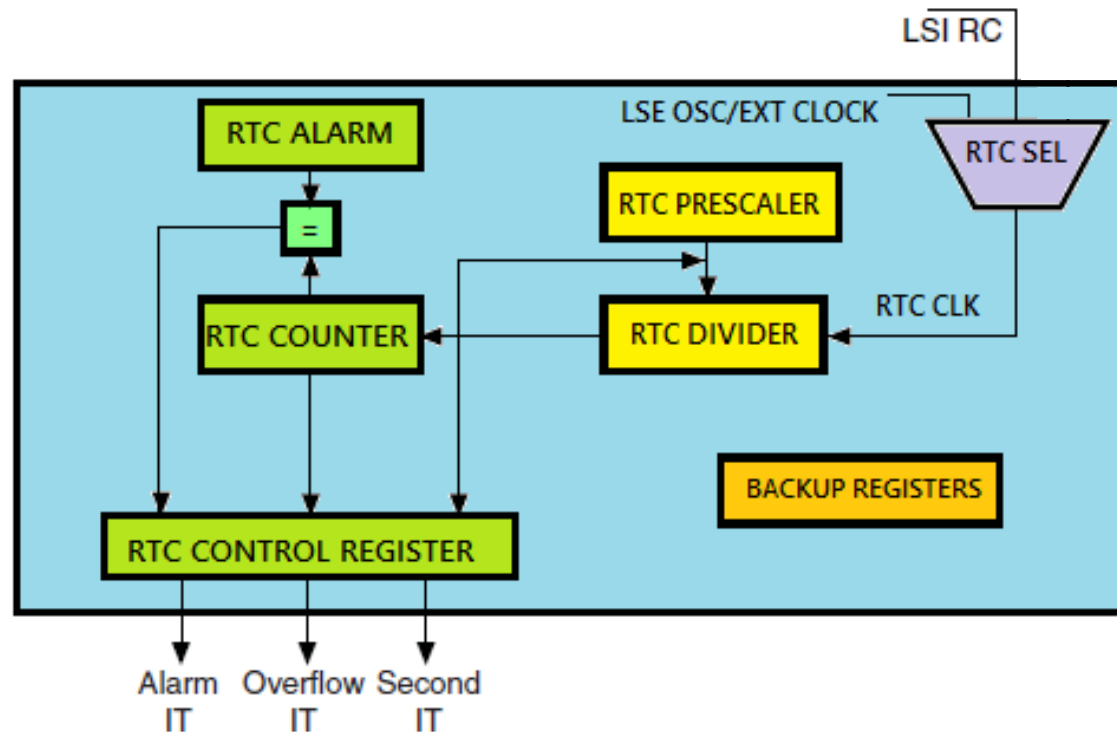
پس از انتخاب منبع، کلاک وارد بخش Pre Scaler می‌شود.



شرح عملکرد واحد RTC

۲. Prescaler & Divider

کلاک ورودی ابتدا توسط RTC Prescaler و سپس توسط RTC Divider تقسیم می‌شود تا در نهایت کلاک ۱ Hz برای شمارنده RTC تولید شود. این کار باعث می‌شود RTC بتواند زمان واقعی را با دقت ثانیه نگه دارد.

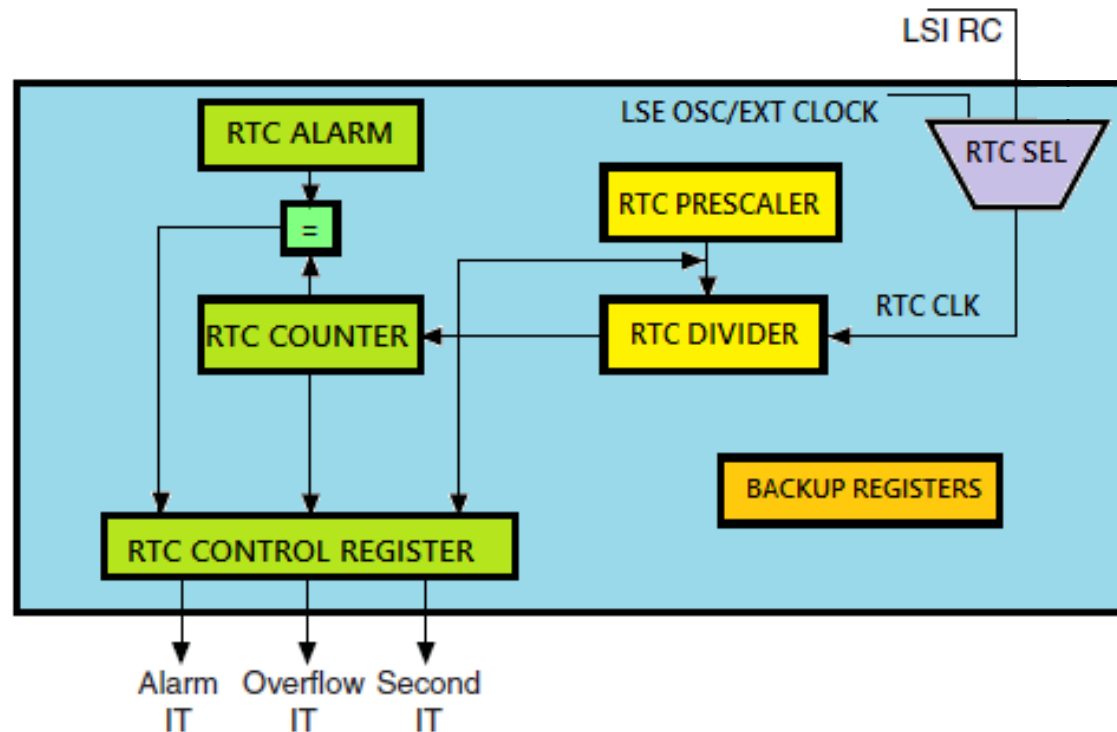


شرح عملکرد واحد RTC

۳. شمارنده (RTC Counter)

این بخش قلب اصلی RTC است و زمان را نگه می‌دارد. وظیفه آن:

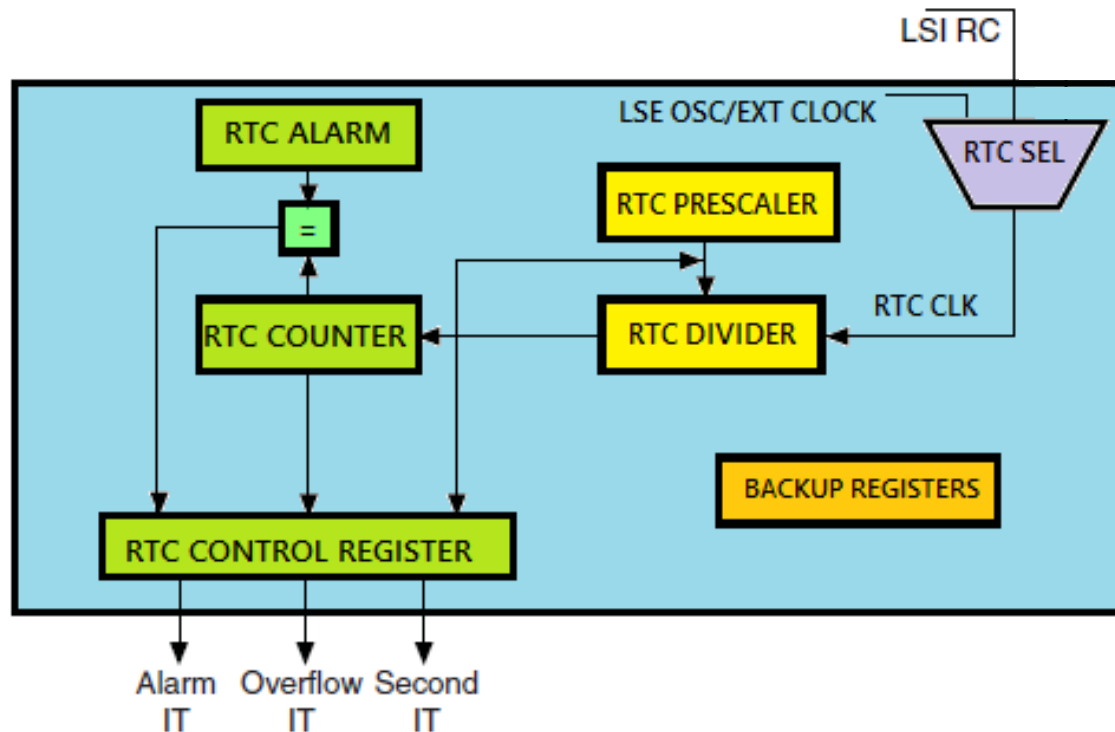
- ✓ افزایش هر ثانیه
- ✓ نگهداری ساعت، دقیقه و ثانیه
- ✓ تولید رخداد Second Interrupt



شرح عملکرد واحد RTC

۴. آلارم (RTC Alarm)

زمان آلارم با شمارنده RTC مقایسه می‌شود. هنگام برابر شدن، وقفه Alarm Interrupt فعال می‌شود و می‌تواند MCU را از حالت کم‌مصرف بیدار کند.

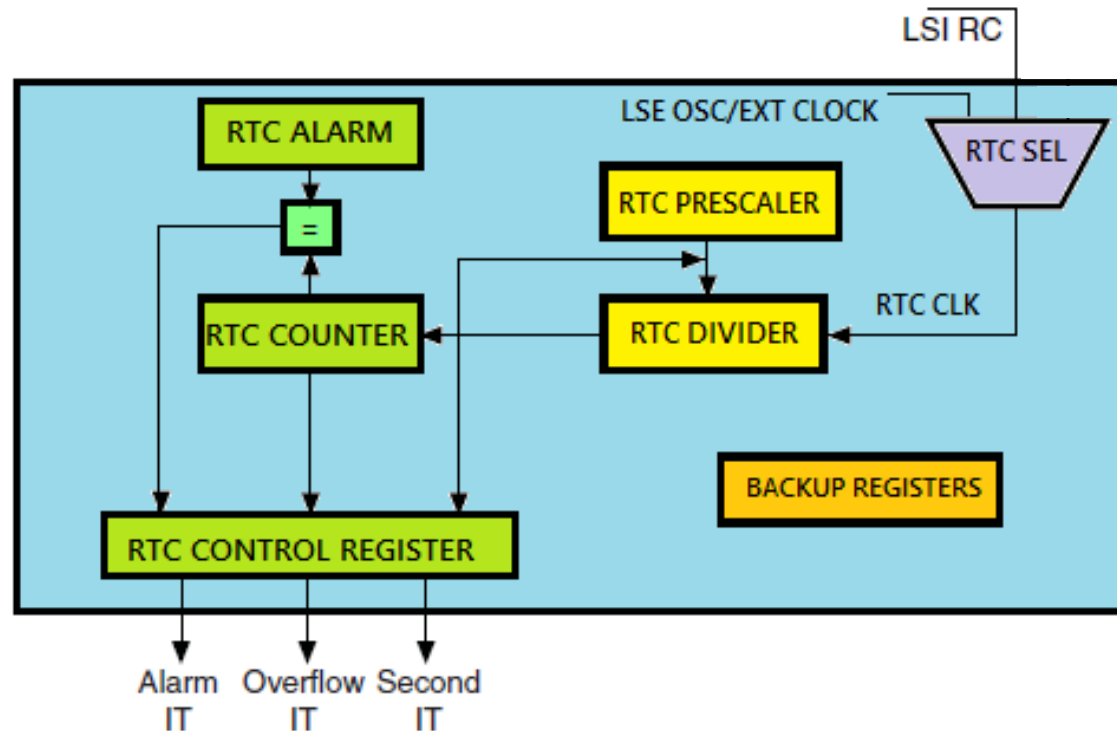


شرح عملکرد واحد RTC

۵. رجیستر کنترل (RTC Control Register)

این بخش کنترل عملکردهای مختلف RTC را برعهده دارد و سه وقفه زیر را مدیریت می‌کند:

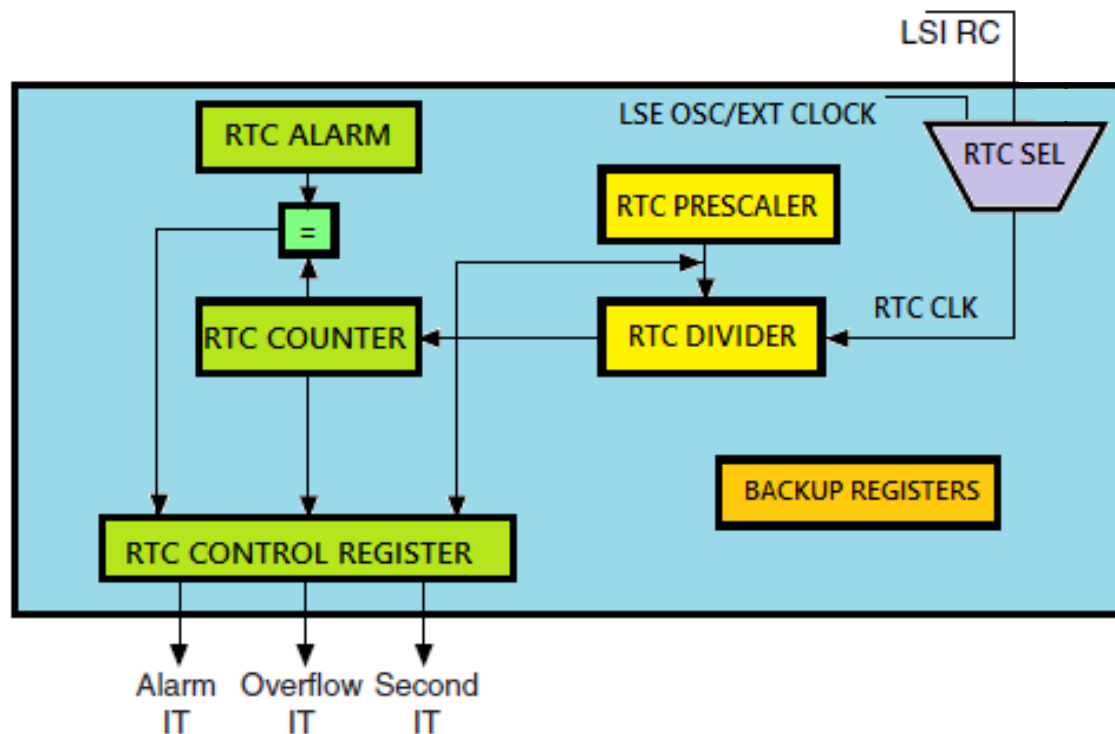
- ✓ Alarm Interrupt
- ✓ Overflow Interrupt
- ✓ Second Interrupt



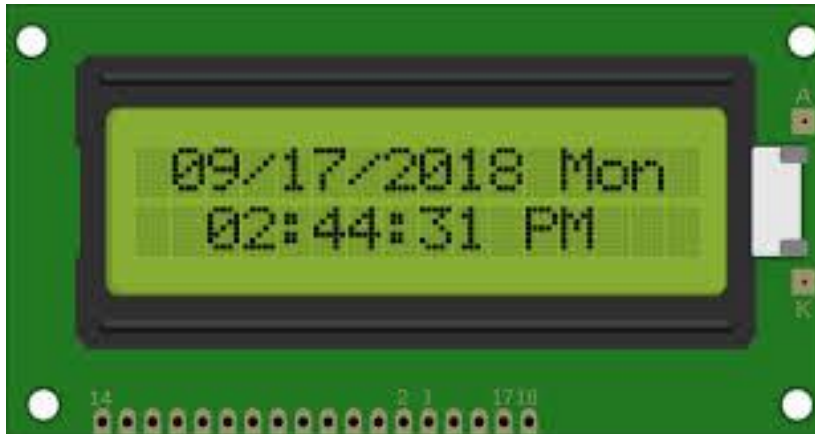
شرح عملکرد واحد RTC

۶. رجیسترهای بکاپ (Backup Registers)

این رجیسترها برای ذخیره داده‌های کاربر استفاده می‌شوند و با وجود تغذیه VBAT حتی پس از ریست شدن سیستم باقی می‌مانند.



مثال کاربردی



در این مثال، میخواهیم از برد توسعه STM32F0Discovery استفاده کنیم:

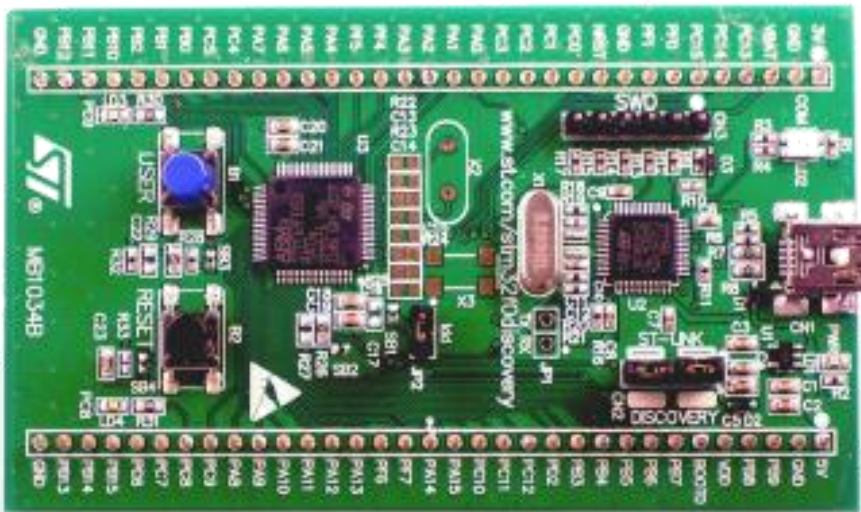
1- در مرحله اول هدف نمایش ساعت، دقیقه و ثانیه بر روی نمایشگر کاراکتری 2 در 16 می باشد.

2- در مرحله دوم تاریخ را در سطر پایین نمایشگر در زیر ساعت نشان دهید.

در مرحله سوم هم یک هشدار را ست نموده و در صورت رخداد، ال ای دی شماره 8 پورت C را روشن نماید. همچنین به جای نمایش تاریخ، زمان آلارم نمایش داده شود.

فرضیات:

✓ برای نمایش زمان از واحد RTC استفاده نمایید.





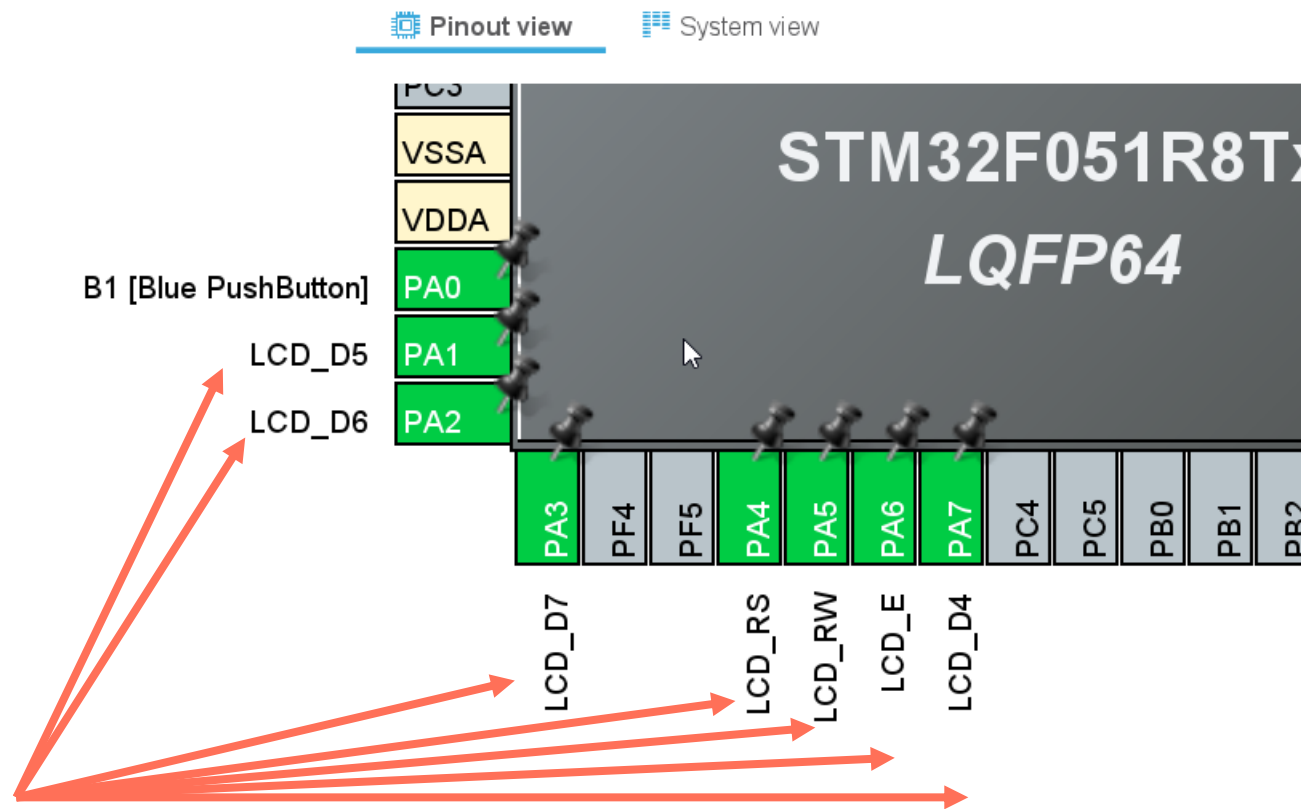
مثال کاربردی واحد RTC

ابتدا کتابخانه LCD کاراکتری را به پروژه اضافه کنید.

- ▼ LCD_F0
 - > Binaries
 - > Includes
 - ▼ Core
 - ▼ Inc
 - > **lcd4bit.h**
 - > main.h
 - > stm32f0xx_hal_conf.h
 - > stm32f0xx_it.h
 - ▼ Src
 - > **lcd4bit.c**
 - > main.c
 - > stm32f0xx_hal_msp.c
 - > stm32f0xx_it.c
 - > syscalls.c
 - > sysmem.c
 - > system_stm32f0xx.c
 - > Startup
 - > Drivers
 - > Debug
 - MX LCD_F0.ioc
 - LCD_F0 Debug.launch
 - STM32F051R8TX_FLASH.ld
 - LED_BLINK (in Class_projects)

مثال کاربردی واحد RTC

پایه های Output جهت راه اندازی LCD کاراکتری را پیکربندی کرده و برچسب های مربوطه را برای هر پایه تعریف کنید.





مثال کاربردی واحد RTC

سرعت تبادل داده پایه های مرتبط با LCD را بر روی High قرار دهید.

GPIO Mode and Configuration

Configuration

Group By Peripherals

GPIO RCC SYS

Search Signals

Search (Ctrl+F)

☐ Show only Modified Pins

Pi...	Signal ...	GPIO ...	GPIO ...	GPIO ...	Maxim...	Fast ...	User L...	Modifi...
PA1	n/a	Low	Output...	No pul...	High	n/a	LCD_...	✓
PA2	n/a	Low	Output...	No pul...	High	n/a	LCD_...	✓
PA3	n/a	Low	Output...	No pul...	High	n/a	LCD_...	✓
PA4	n/a	Low	Output...	No pul...	High	n/a	LCD_...	✓
PA5	n/a	Low	Output...	No pul...	High	n/a	LCD_...	✓
PA6	n/a	Low	Output...	No pul...	High	n/a	LCD_E	✓
PA7	n/a	Low	Output...	No pul...	High	n/a	LCD_...	✓

GPIO mode: Output Push Pull

GPIO Pull-up/Pull-down: No pull-up and no pull-down

Maximum output speed: High

User Label: LCD_RW

Pinout view System view

System Core Analog

DMA

GPIO ✓

NVIC ✓

RCC ✓

SYS ✓



مثال کاربردی واحد RTC

جهت تست عملکرد LCD ابتدا یک بار کد را تولید نموده و به main.c مربوطه مراجعه نمایید.

LCD_F0.ioc - Pinout & Configuration

Pinout & Configuration | Clock Configuration | Project Manager | Tools

Software Packs | Pinout

GPIO Mode and Configuration

Configuration

Group By Peripherals

GPIO | RCC | SYS

Search Signals

Search (Ctrl+F)

☐ Show only Modified Pins

Do you want to generate Code?

☐ Remember my decision

Yes No

System Core | Analog

DMA

GPIO

NVIC

RCC

PA7	n/a	Low	Output...	No pul...	High	n/a	LCD_...
GPIO mode			Output Push Pull				
GPIO Pull-up/Pull-down			No pull-up and no pull-down				



مثال کاربردی واحد RTC

کتابخانه‌های لازم جهت کار با کتابخانه را در کد اضافه کنید.

```
/* Private includes -----*/  
/* USER CODE BEGIN Includes */  
#include "stdio.h"  
#include "lcd4bit.h"  
/* USER CODE END Includes */
```

مثال کاربردی واحد RTC

جهت نمایش یک متن تست روی LCD و اطمینان از صحت عملکرد آن کد زیر را به main.c افزوده و میکرو را پروگرام نمایید.

```
/* USER CODE BEGIN WHILE */
lcd_init();
lcd_gotoxy(6,0);
lcd_hide_cursor();
lcd_puts("IUST");
lcd_gotoxy(1,1);
lcd_puts("uProcessor LAB");
HAL_Delay(2000);
lcd_clear();
while (1)
{
/* USER CODE END WHILE */

/* USER CODE BEGIN 3 */
}
/* USER CODE END 3 */
```



مثال کاربردی واحد RTC

پس از اطمینان از عملکرد LCD حال برای تنظیمات و راه اندازی واحد RTC طبق تصویر زیر عمل نمایید.

1 Categories

2 RTC

3 Activate Clock Source

4 Activate Calendar

5 Alarm A Internal Alarm A

RTC Mode and Configuration

Mode

☒ Activate Clock Source

☒ Activate Calendar

Alarm A Internal Alarm A

☐ Timestamp

☐ Tamper 1

☒ Tamper 2

Calibration Disable

☐ Reference clock detection

Configuration

Reset Configuration

☒ Parameter Settings ☒ User Constants ☒ NVIC Settings

Configure the below parameters :

Search (Ctrl+F)

General

Hour Format Hourformat 24

Asynchronous Predivider val.. 127



مثال کاربردی واحد RTC

Pinout & Configuration | Clock Configuration | Project Manager

Software Packs | Pinout

Categories | A->Z

- ✓ GPIO
- HDMI_CEC
- I-CUBE-FS-RTOS
- I-CUBE-ITTIADB
- I-CUBE-Mongoose
- I-CUBE-embOS
- I-CUBE-wolfMQTT
- I-CUBE-wolfSSH
- I-CUBE-wolfSSL
- I-CUBE-wolfTPM
- I2C1
- I2C2
- I2S1
- IRTIM
- IWDG
- ✓ NVIC
- ✓ RCC
- ⚠ RTC
- SPI1
- SPI2
- ⚠ SYS
- TIM1
- TIM2
- TIM3
- TIM6

RTC Mode and Configuration

Mode

- ✓ Activate Clock Source
- ✓ Activate Calendar
- Alarm A: Internal Alarm A

Configuration

Reset Configuration

Parameter Settings | User Constants | NVIC Settings

Configure the below parameters :

Search (Ctrl+F)

General

- Hour Format: Hourformat 24
- Asynchronous Predivider val.: 127
- Synchronous Predivider valu.: 255

Calendar Time

- Data Format: Binary data format
- Hours: 10
- Minutes: 59
- Seconds: 37
- Day Light Saving: value of ho. Daylightsaving None
- Store Operation: Storeoperation Reset

Calendar Date

6



مثال کاربردی واحد RTC

Pinout & Configuration | Clock Configuration | Project Manager

Software Packs | Pinout

Categories | A->Z

- GPIO
- HDMI_CEC
- I-CUBE-FS-RTOS
- I-CUBE-ITTIADB
- I-CUBE-Mongoose
- I-CUBE-embOS
- I-CUBE-wolfMQTT
- I-CUBE-wolfSSH
- I-CUBE-wolfSSL
- I-CUBE-wolfTPM
- I2C1
- I2C2
- I2S1
- IRTIM
- IWDG
- ✓ NVIC
- ✓ RCC
- ⚠ RTC
- SPI1
- SPI2
- ⚠ SYS
- TIM1
- TIM2
- TIM3
- TIM6

RTC Mode and Configuration

Mode

- ✓ Activate Clock Source
- ✓ Activate Calendar
- Alarm A: Internal Alarm A

Configuration

Reset Configuration

Parameter Settings | User Constants | NVIC Settings

Configure the below parameters :

Search (Ctrl+F)

Alarm A	
Hours	11
Minutes	14
Seconds	45
Sub Seconds	0
Alarm Mask Date Week day	Enable
Alarm Mask Hours	Disable
Alarm Mask Minutes	Disable
Alarm Mask Seconds	Disable
Alarm Sub Second Mask	All Alarm SS fields are masked.
Alarm Date Week Day Sel	Date



مثال کاربردی واحد RTC

وقفه آلارم RTC را فعال کنید.

Configuration | Clock Configuration | Project Manager | Tools

Software Packs | Pinout

NVIC Mode and Configuration

Mode

Configuration

✓ NVIC | ✓ Code generation

☐ Sort by Preemption Priority and Sub Priority ☐ Sort by

Search [Se...] Show [available interrupts] ☒ Force

NVIC Interrupt Table	Enabled
Non maskable interrupt	☒
Hard fault interrupt	☒
System service call via SWI instruction	☒
Pendable request for system service	☒
Time base: System tick timer	☒
PVD interrupt through EXTI line16	☐
RTC global interrupt through EXTI lines 17, 19 and 20	☒
Flash global interrupt	☐
RCC global interrupt	☐

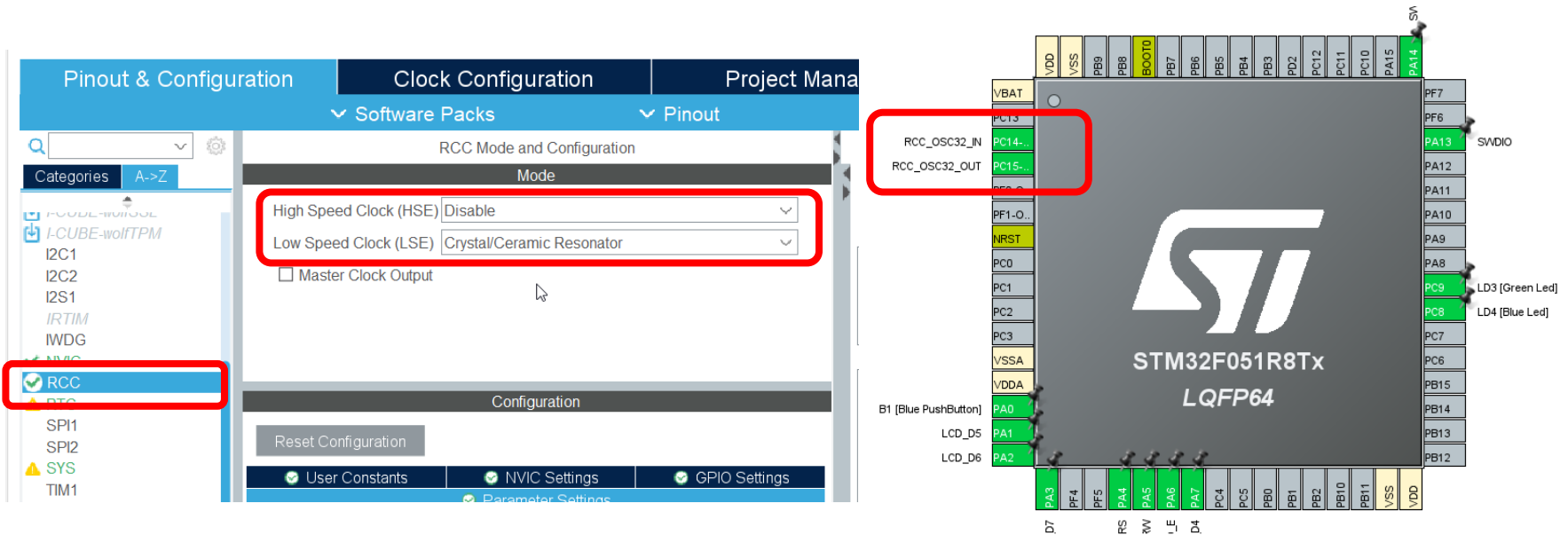
Pinout view | **System view**

System Core | Analog | Timers

DMA | GPIO ✓ | **NVIC ✓** | RCC ✓ | SYS ✓

RTC ✓

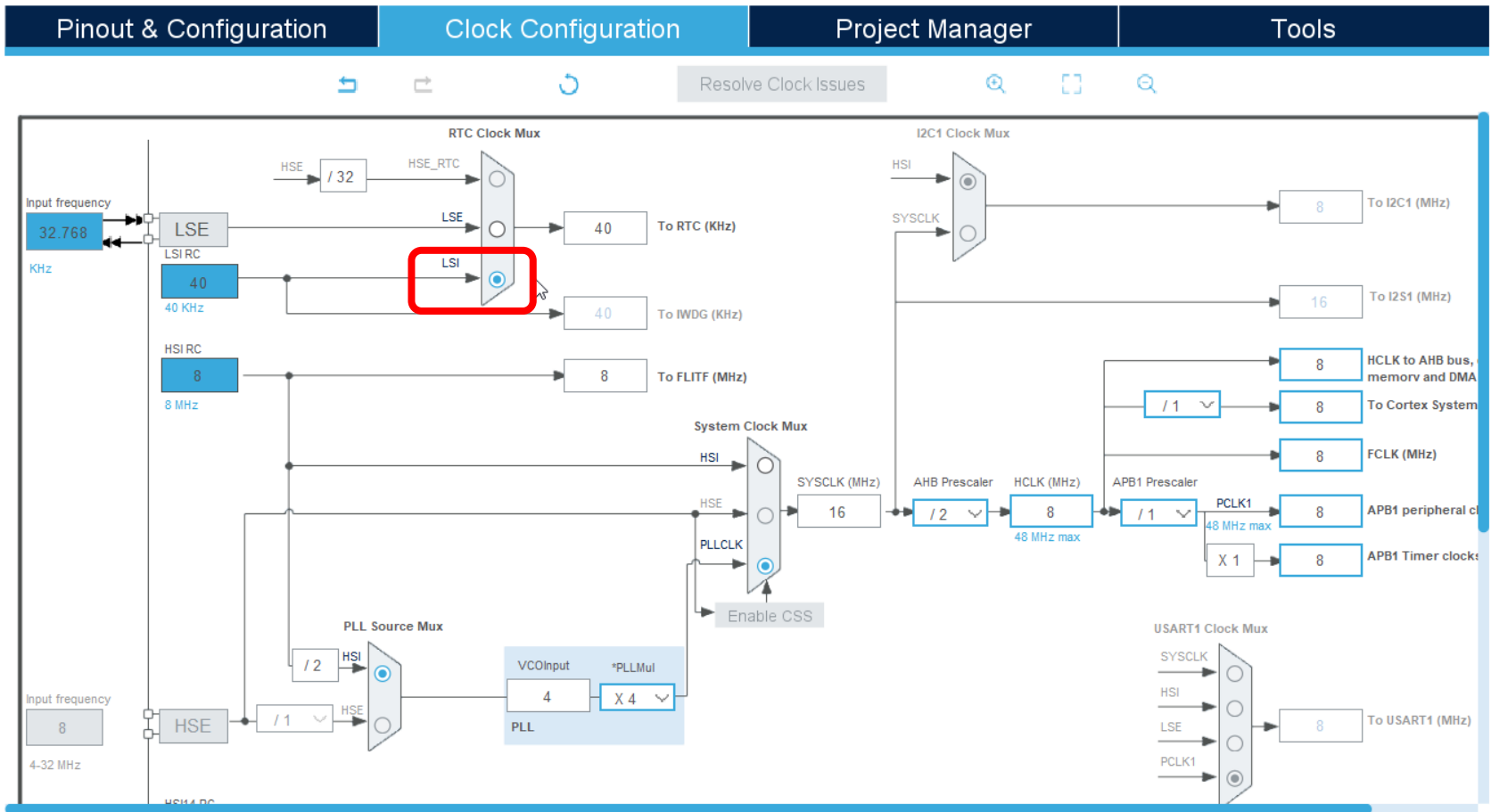
منبع کلاک مناسب را انتخاب نمایید.





مثال کاربردی واحد RTC

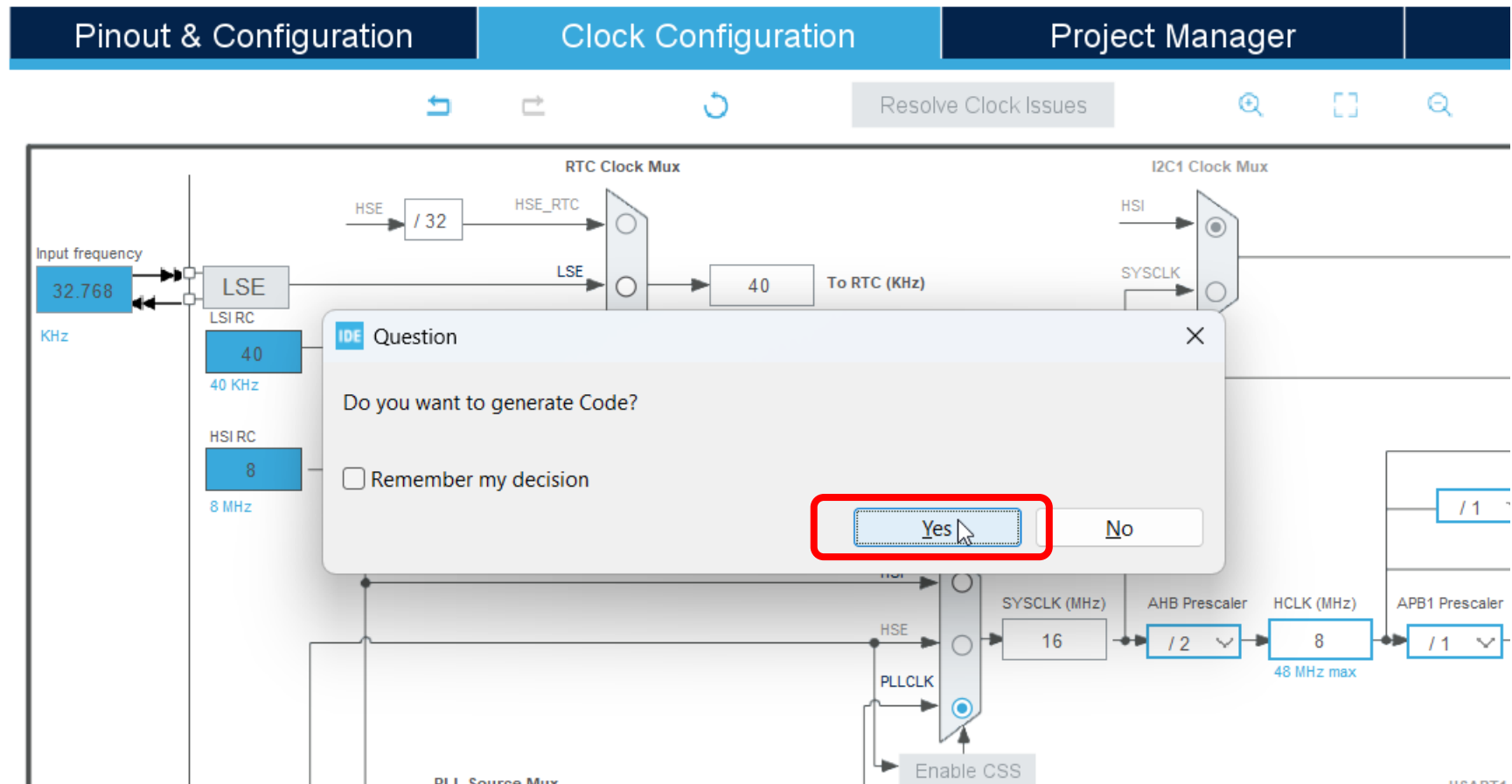
تنظیمات کلاک را انجام دهید.





مثال کاربردی واحد RTC

کد تولید شده را بروزرسانی نمایید.



مثال کاربردی واحد RTC

```
/* USER CODE BEGIN PV */  
char buffer_1[20];
```

برای خواندن زمان، تاریخ و آلارم به سه استراکچر نیاز داریم:

```
RTC_TimeTypeDef gTime = {0};  
RTC_DateTypeDef gDate = {0};  
RTC_AlarmTypeDef gAlarm = {0};  
/* USER CODE END PV */
```

مثال کاربردی واحد RTC

جهت نمایش ساعت از کدهای زیر استفاده نمایید.

```
while (1)
{
    HAL_RTC_GetTime(&hrtc, &gTime, RTC_FORMAT_BIN);
    HAL_RTC_GetDate(&hrtc, &gDate, RTC_FORMAT_BIN);
    HAL_RTC_GetAlarm(&hrtc, &gAlarm, RTC_ALARM_A, RTC_FORMAT_BIN);
    sprintf(buffer_1, "Time:%02u:%02u:%02u", gTime.Hours, gTime.Minutes, gTime.Seconds);
    lcd_gotoxy(0,0);
    lcd_puts(buffer_1);
    HAL_Delay(200);

    /* USER CODE END WHILE */

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


مثال کاربردی واحد RTC

جهت نمایش تاریخ از کدهای زیر استفاده نمایید.

```
while (1)
{
    HAL_RTC_GetTime(&hrtc, &gTime, RTC_FORMAT_BIN);
    HAL_RTC_GetDate(&hrtc, &gDate, RTC_FORMAT_BIN);
    HAL_RTC_GetAlarm(&hrtc, &gAlarm, RTC_ALARM_A, RTC_FORMAT_BIN);
    sprintf(buffer_1, "Time:%02u:%02u:%02u", gTime.Hours, gTime.Minutes, gTime.Seconds);
    lcd_gotoxy(0,0);
    lcd_puts(buffer_1);
    sprintf(buffer_1, "Date:%02u/%02u/%02u %2u", gDate.Year, gDate.Month, gDate.Date, gDate.WeekDay);
    lcd_gotoxy(0,1);
    lcd_puts(buffer_1);
    HAL_Delay(200);

    /* USER CODE END WHILE */

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



مثال کاربردی واحد RTC

جهت نمایش آلارم از کدهای زیر استفاده نمایید.

```
while (1)
{
    HAL_RTC_GetTime(&hrtc, &gTime, RTC_FORMAT_BIN);
    HAL_RTC_GetDate(&hrtc, &gDate, RTC_FORMAT_BIN);
    HAL_RTC_GetAlarm(&hrtc, &gAlarm, RTC_ALARM_A, RTC_FORMAT_BIN);
    sprintf(buffer_1, "Time:%02u:%02u:%02u", gTime.Hours, gTime.Minutes, gTime.Seconds);
    lcd_gotoxy(0,0);
    lcd_puts(buffer_1);
    sprintf(buffer_1, "Alarm:%02u:%02u:%02u", gAlarm.AlarmTime.Hours, gAlarm.AlarmTime.Minutes, gAlarm.AlarmTime.Seconds);
    lcd_gotoxy(0,1);
    lcd_puts(buffer_1);
    HAL_Delay(200);

    /* USER CODE END WHILE */

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

مثال کاربردی واحد RTC

برای فعال کردن وقفه آلارم واحد RTC از کدهای زیر استفاده نمایید.

```
/* USER CODE BEGIN 4 */  
void HAL_RTC_AlarmAEventCallback(RTC_HandleTypeDef *hrtc)  
{  
    /* Prevent unused argument(s) compilation warning */  
    UNUSED(hrtc);  
    /* NOTE : This function should not be modified, when the  
    callback is  
    needed,  
    the HAL_RTC_AlarmAEventCallback could be implemented in  
    the user file  
    */  
    HAL_GPIO_WritePin(GPIOC, GPIO_PIN_8, GPIO_PIN_SET);  
}  
/* USER CODE END 4 */
```



razeghizade@gmail.com

Razeghizade.pudica.ir

CREADITS: This presentation was created by M.Razeghizadeh
Please keep this slide for attribution