In The Name of God

Faculty of Electrical Engineering

Microprocessor Laboratory

V1 (@2025)

Microprocessor LAB

Lecture 7:

Internal RTC in STM32

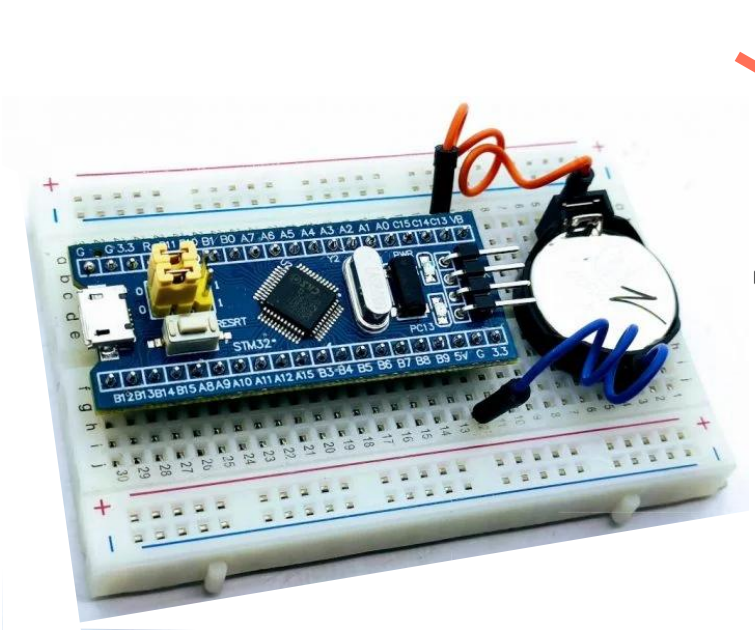
By: M.Razeghizadeh

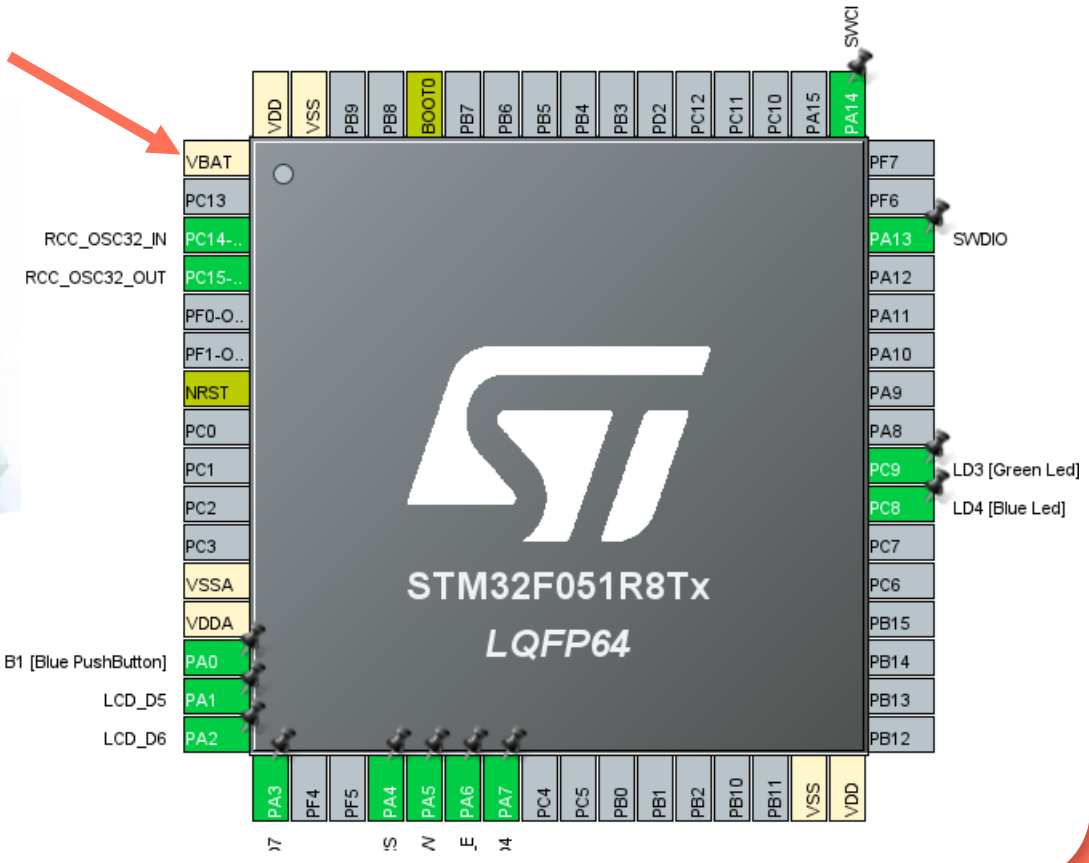
Start now!

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

RTC چیست؟ (Real-Time Clock)

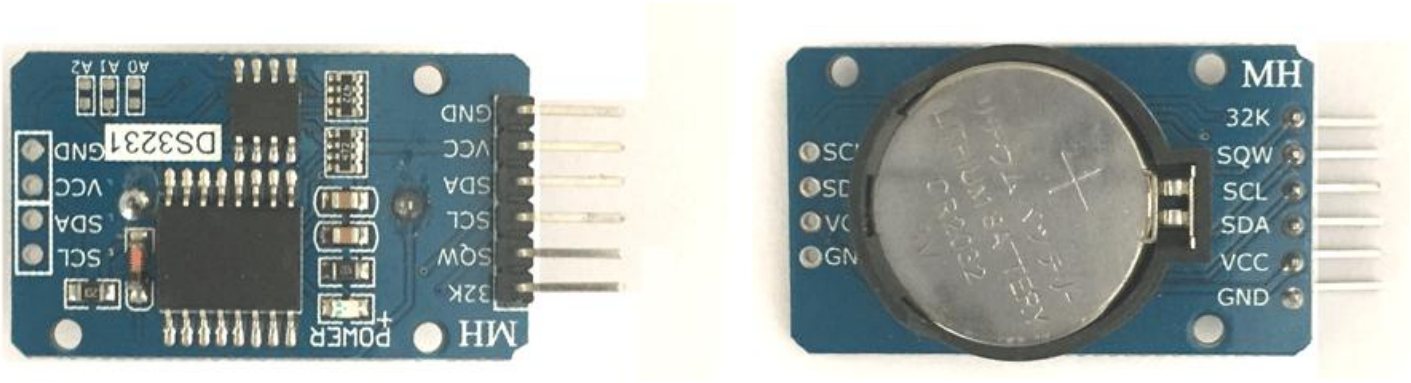
- واحد RTC در میکروکنترلرهای STM32 یک محیط سخت‌افزاری اختصاصی برای نگه‌داری زمان و تاریخ است؛ حتی زمانی که تغذیهٔ اصلی قطع شود. این واحد با کمک منبع تغذیهٔ پشتیبان VBAT می‌تواند اطلاعات زمان را برای مدت طولانی حفظ کند.





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 تغذیهٔ 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 به‌صورت قدم‌به‌قدم انجام می‌دهیم:

- راه‌اندازی و پیکربندی RTC داخلی
- انتخاب منبع کلاک مناسب (LSIیا LSE)
- 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 وظیفهٔ 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، به‌صورت *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 گزینهٔ 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

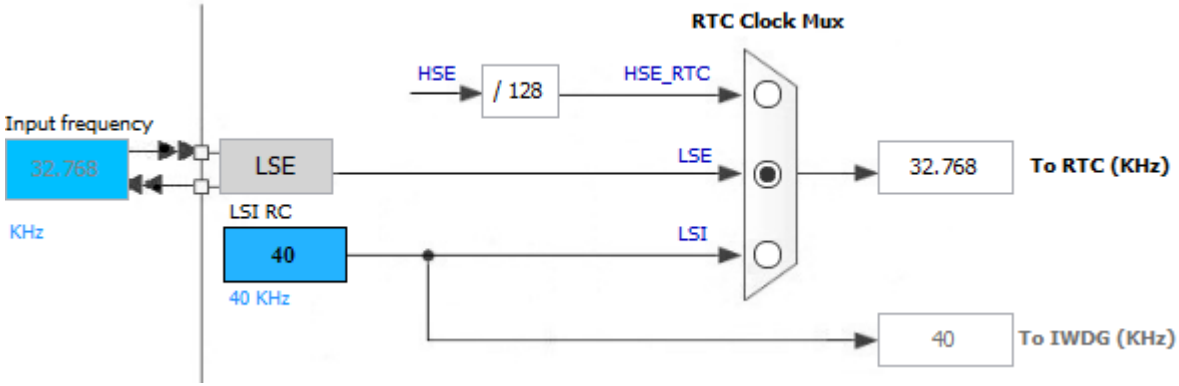
- **انتخاب بین LSE و LSI به نیازهای پروژه بستگی دارد:**

- **انتخاب LSE:**
  - ✓ اگر دقت زمان واقعی حیاتی است. (مثلاً ساعت دقیق، ثبت رویدادهای حساس به زمان)
  - ✓ اگر هدف کارکرد طولانی‌مدت و پایدار باشد.
  - ✓ وقتی فضای برد و هزینه اضافه برای کریستال خارجی مشکلی ایجاد نمی‌کند.

- **انتخاب LSI:**
  - ✓ اگر دقت خیلی بالا مهم نیست و کمی خطا یا drift قابل قبول است.
  - ✓ برای پروژه‌های کم‌هزینه یا نمونه اولیه.
  - ✓ وقتی فضای برد محدود است و نمی‌خواهید قطعات خارجی اضافه کنید.

- **به زبان ساده: برای دقت بالا و پروژه‌های جدی LSE، برای سادگی و صرفه‌جویی LSI.**

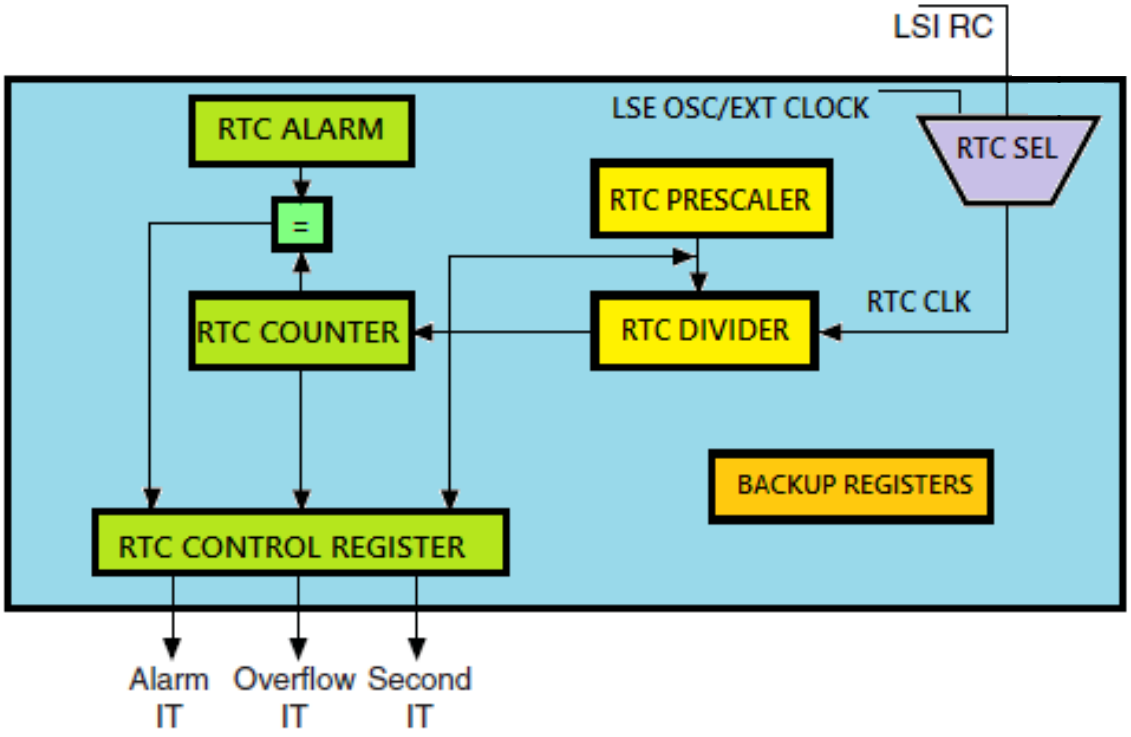
معیار انتخاب منبع کلاک



ویژگی	LSE(کریستال خارجی کم‌سرعت)	LSI(اسیلاتور داخلی کم‌سرعت)
فرکانس	۳۲٫۷۶۸ کیلوهرتز (کریستال استاندارد ساعت)	تقریباً ۴۰ کیلوهرتز (RCداخلی)
دقت	بسیار بالا و پایدار در طول زمان	پایین‌تر، ممکن است با دما و ولتاژ دچار انحراف شود
سخت‌افزار مورد نیاز	کریستال خارجی ۳۲٫۷۶۸ کیلوهرتز	بدون نیاز به سخت‌افزار خارجی
مصرف انرژی	کم	کم
هزینه	کمی بالاتر (به دلیل کریستال اضافی)	بدون هزینه اضافه
مناسب برای	ساعت‌های دقیق، ثبت داده، اینترنت اشیاء (IoT)	آلارم‌های ساده، پروژه‌های کم‌هزینه

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.



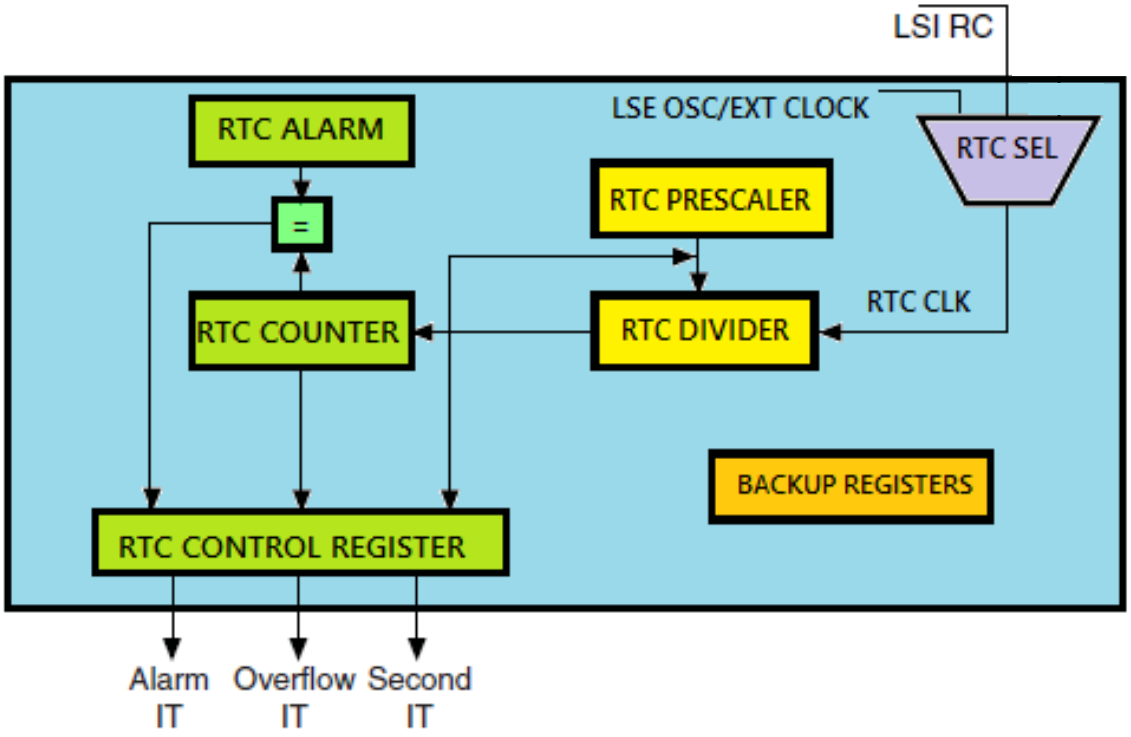
شرح عملکرد واحد RTC

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.

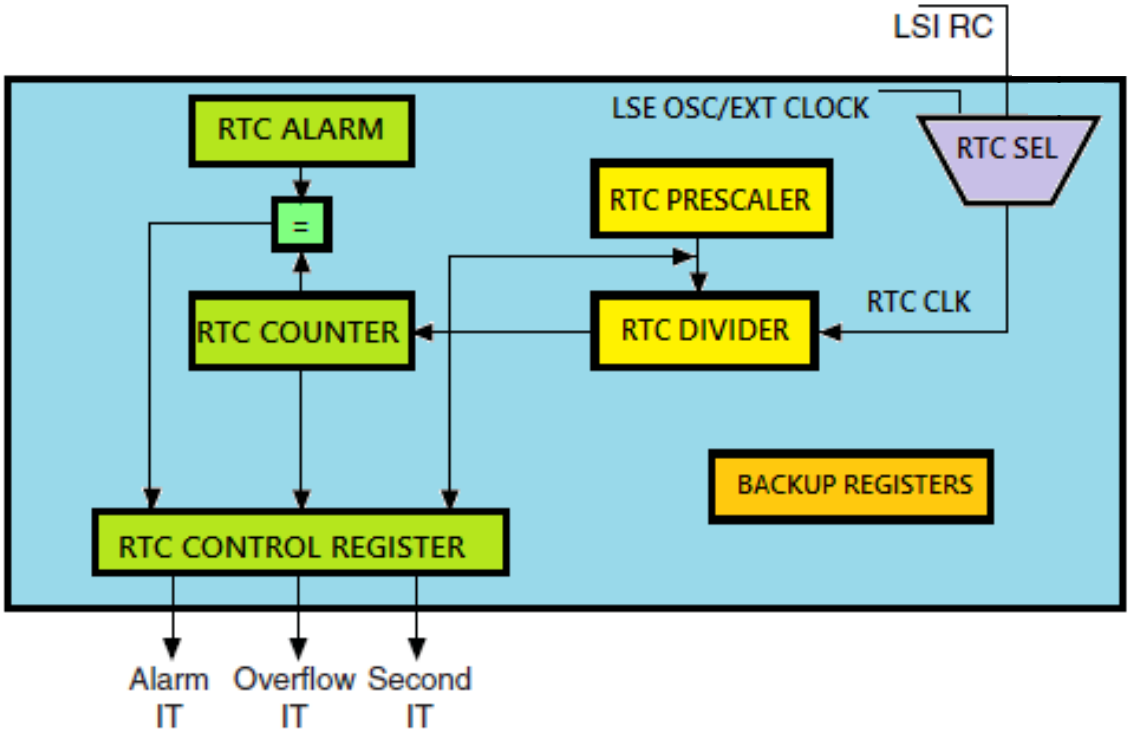


شرح عملکرد واحد RTC

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.

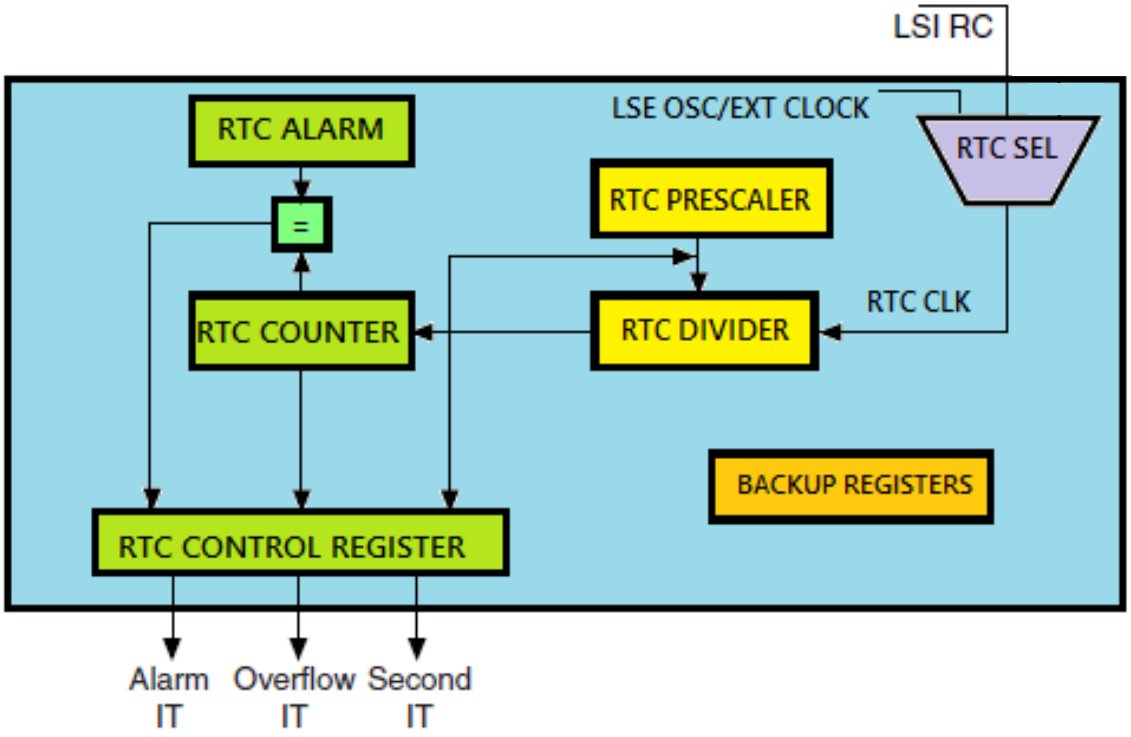


شرح عملکرد واحد RTC

۳. شمارنده RTC (RTC Counter)

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

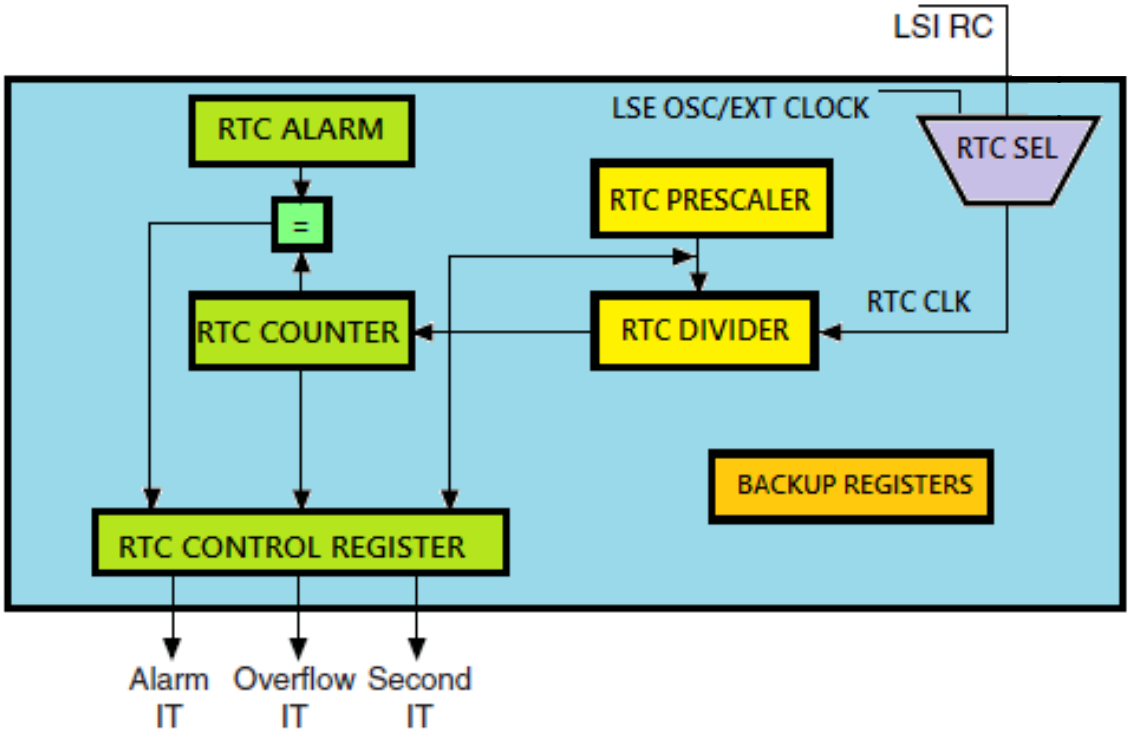
- ✓ افزایش هر ثانیه
- ✓ نگهداری ساعت، دقیقه و ثانیه
- ✓ تولید رخداد Second Interrupt



شرح عملکرد واحد RTC

۴. آلارم RTC (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.

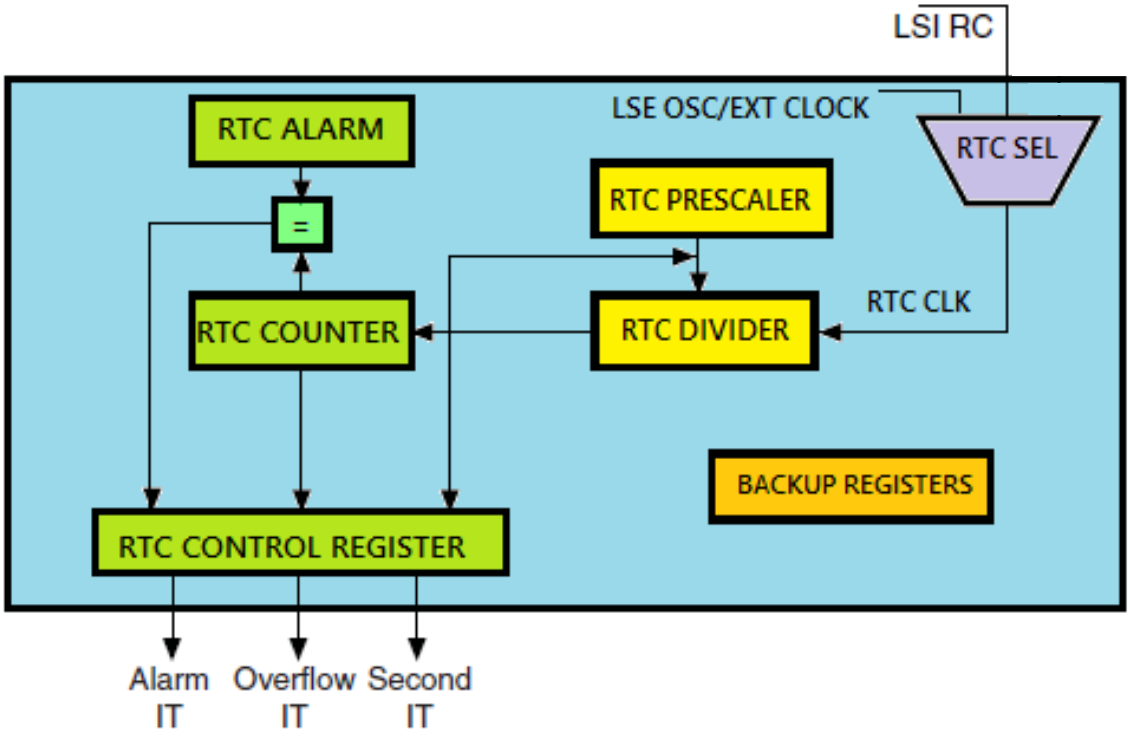


شرح عملکرد واحد RTC

۵. رجیستر کنترل RTC (RTC Control Register)

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

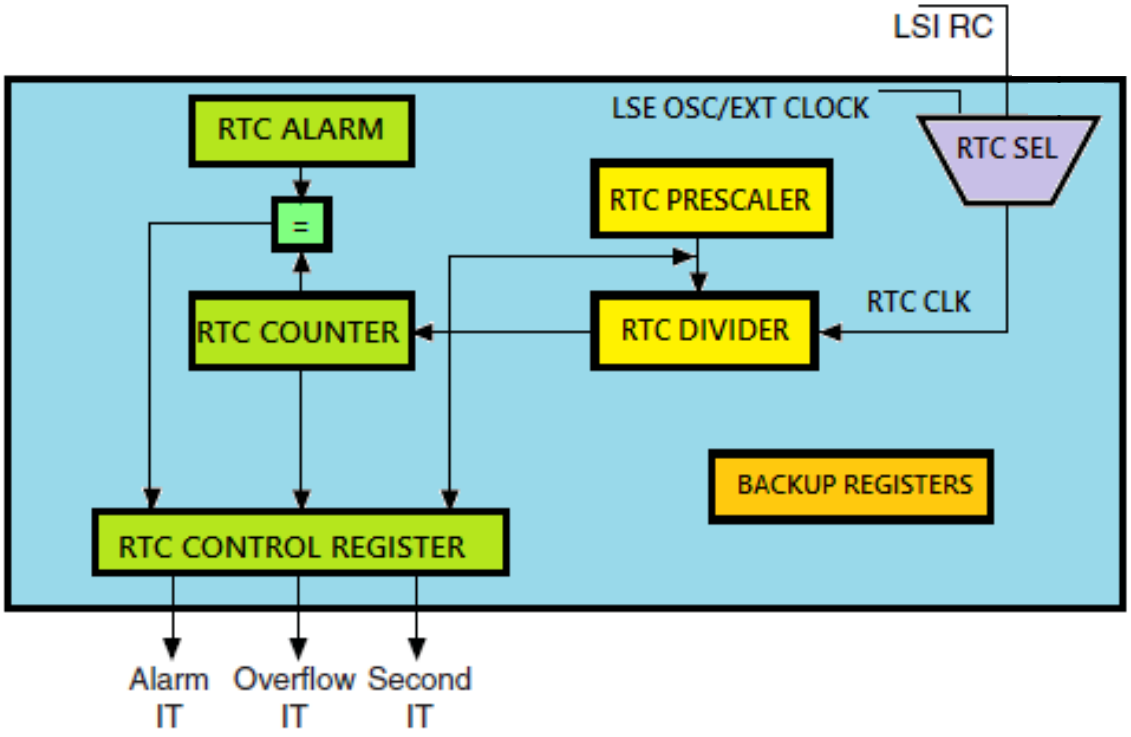
- ✓ Alarm Interrupt
- ✓ Overflow Interrupt
- ✓ Second Interrupt



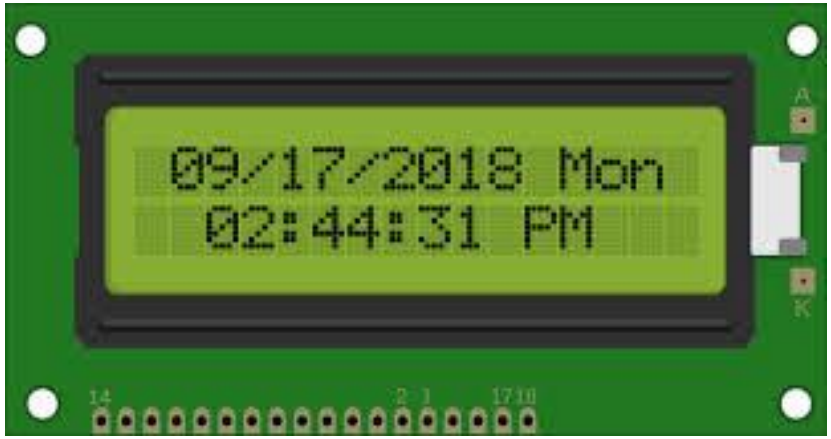
شرح عملکرد واحد RTC

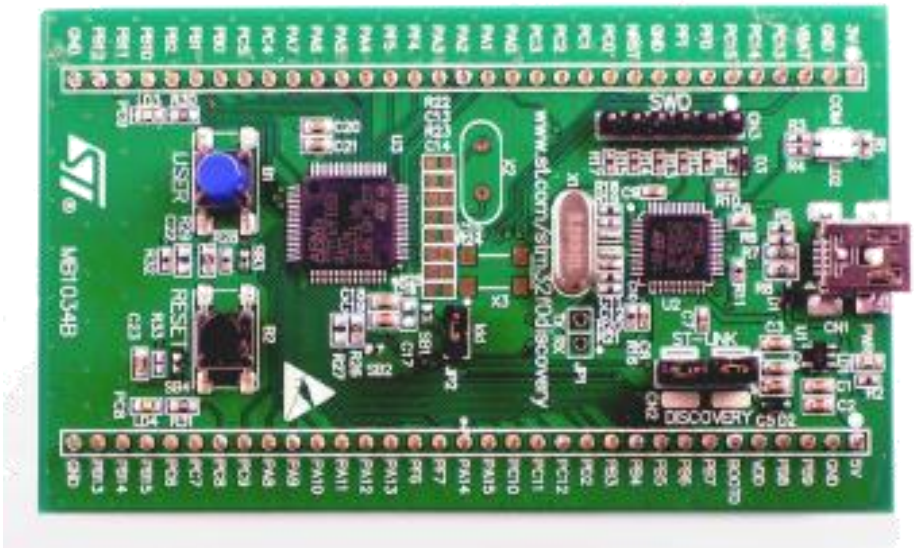
۶. رجیسترهای بکاپ (Backup Registers)

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 می‌باشد.

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

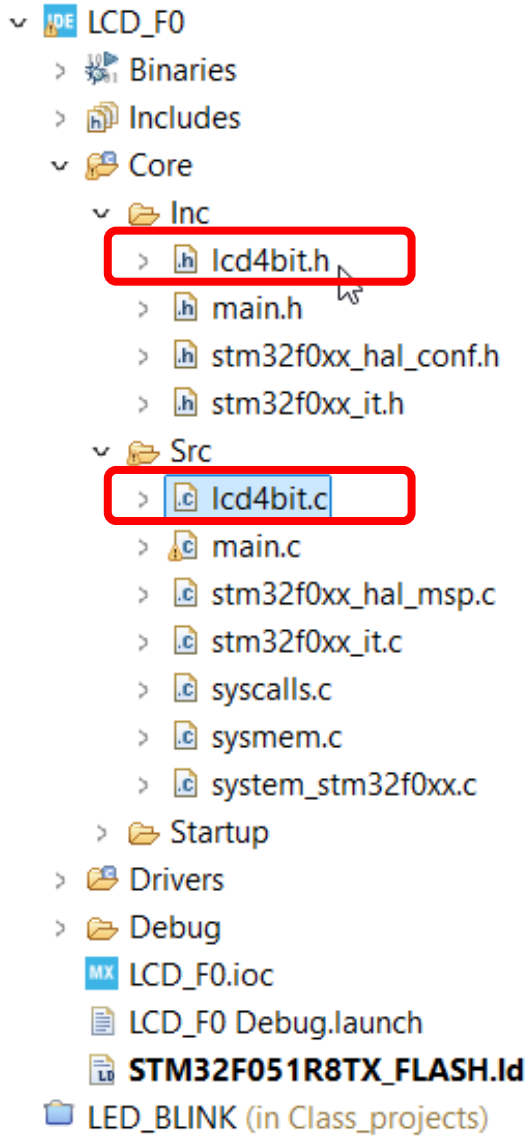
در مرحله سوم هم یک هشدار را ست نموده و در صورت رخداد، ال ای دی شماره 8 پورت C را روشن نماید. همچنین به جای نمایش تاریخ، زمان آلارم نمایش داده شود.

فرضیات:

- ✓ برای نمایش زمان از واحد RTC استفاده نمایید.

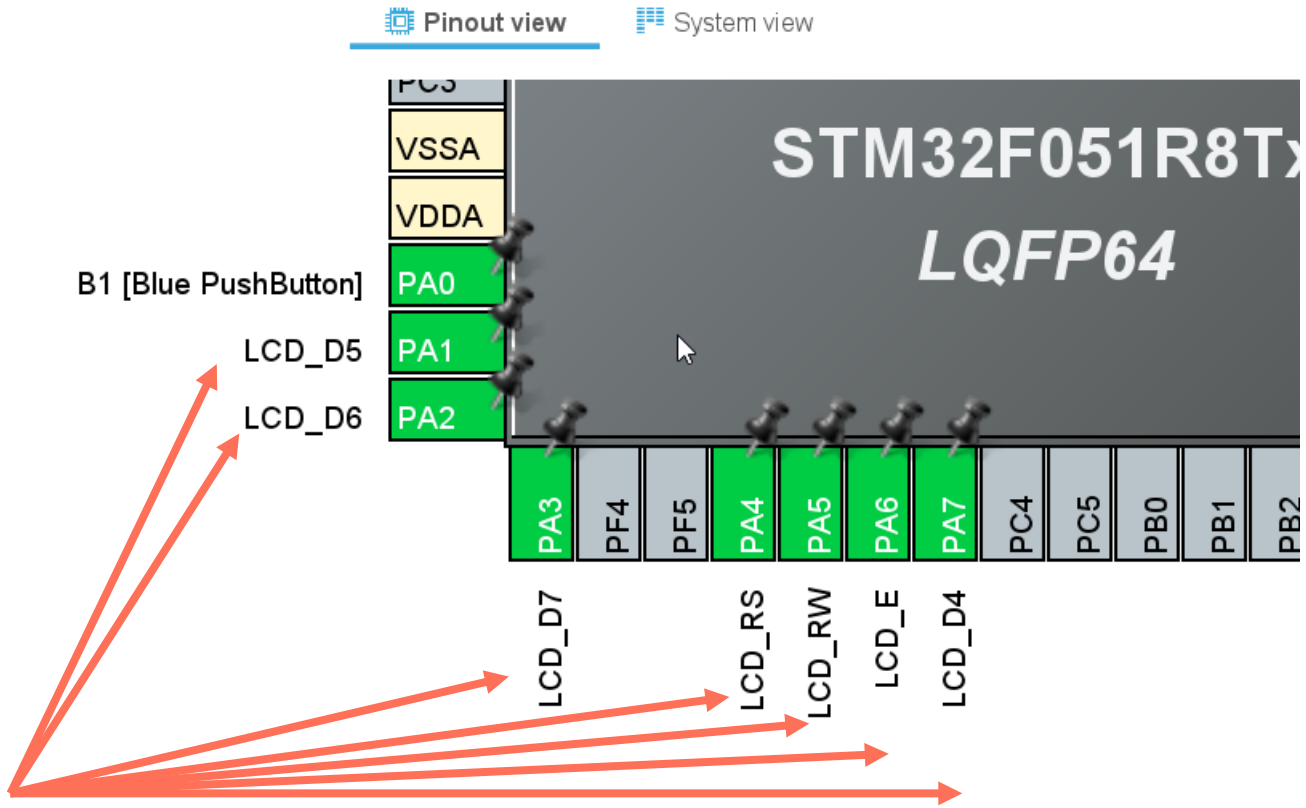
مثال کاربردی واحد RTC

ابتدا کتابخانه LCD کاراکتری را به پروژه اضافه کنید.



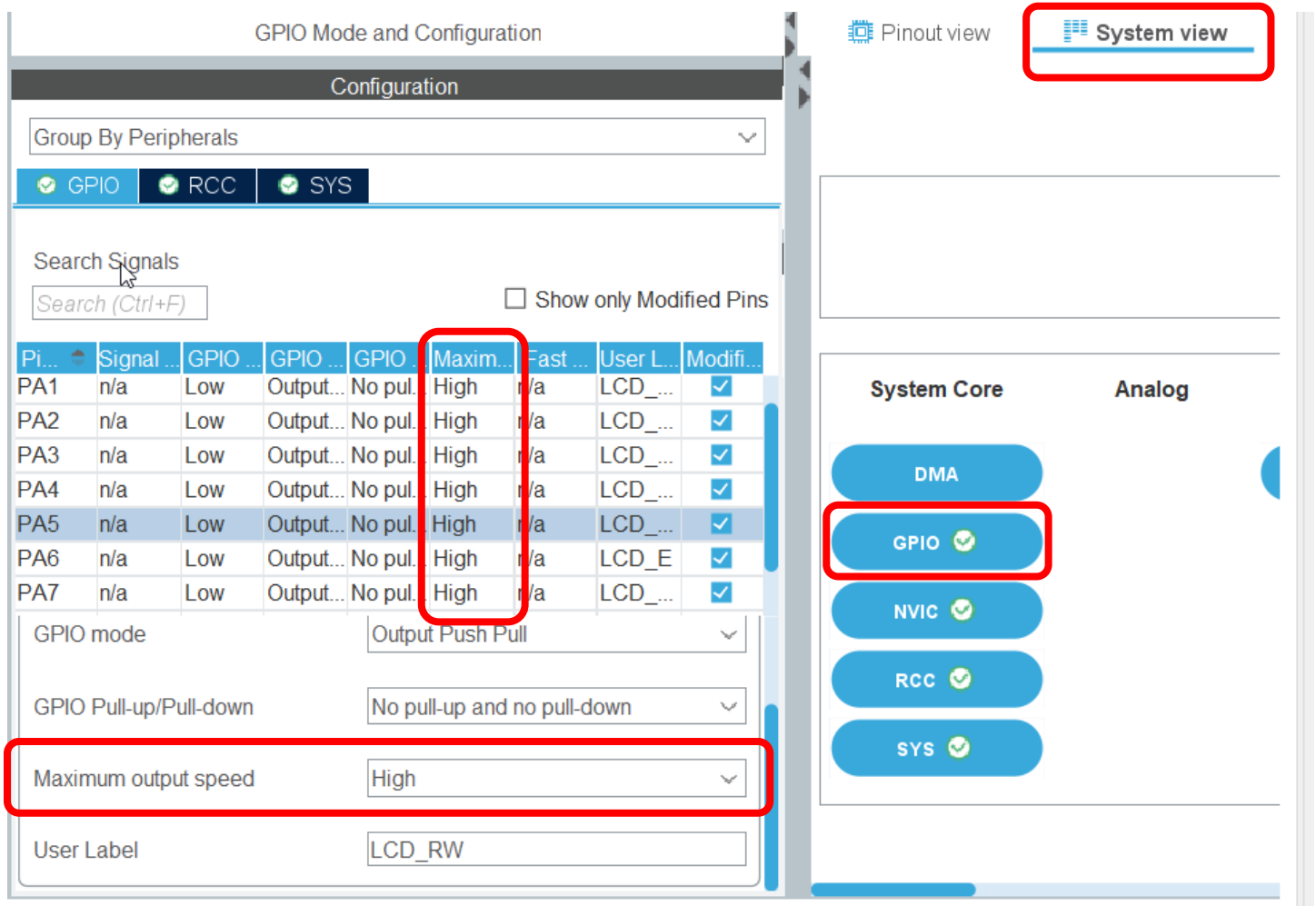
مثال کاربردی واحد RTC

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



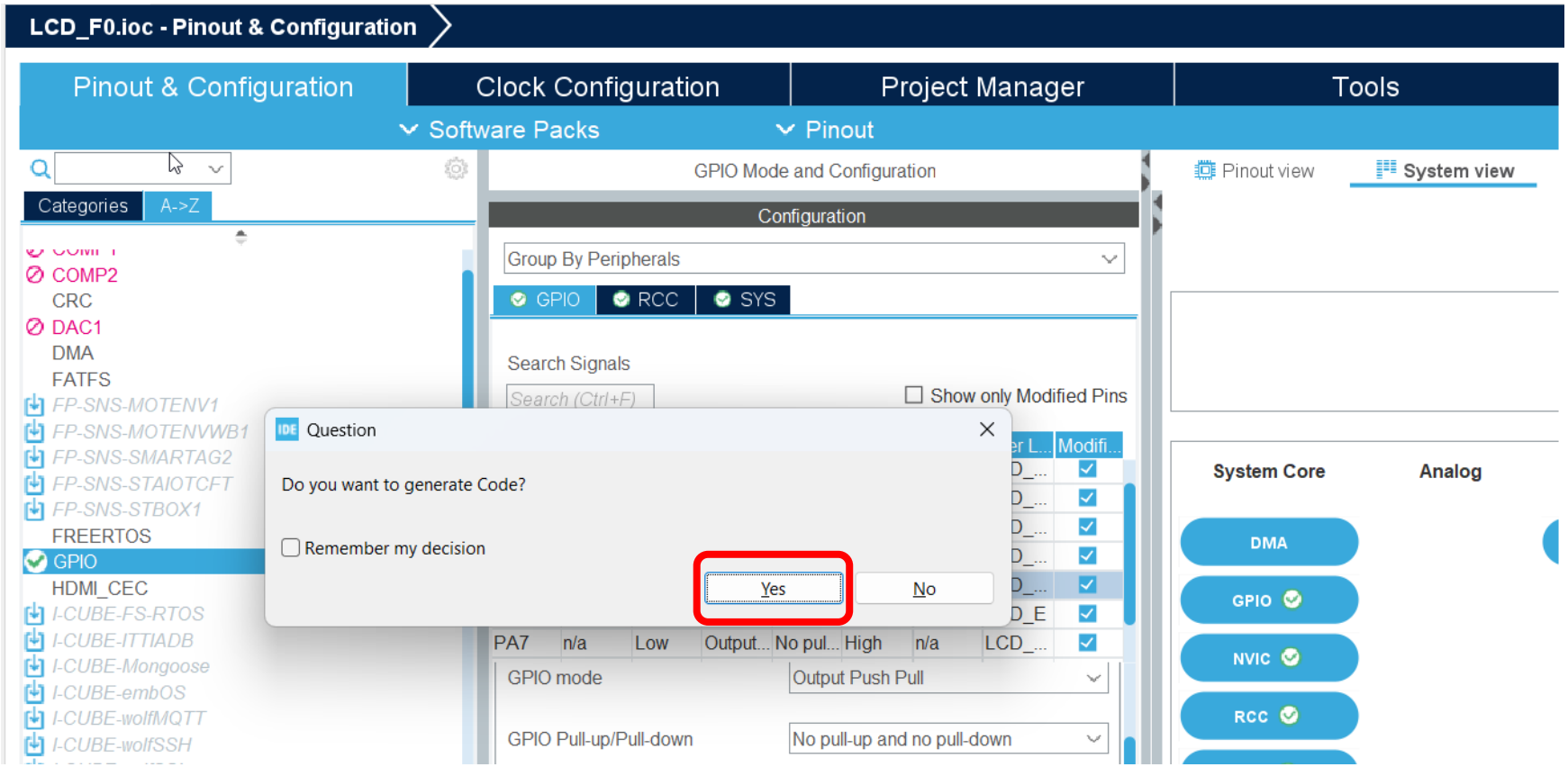
مثال کاربردی واحد RTC

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



مثال کاربردی واحد RTC

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



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 */
```

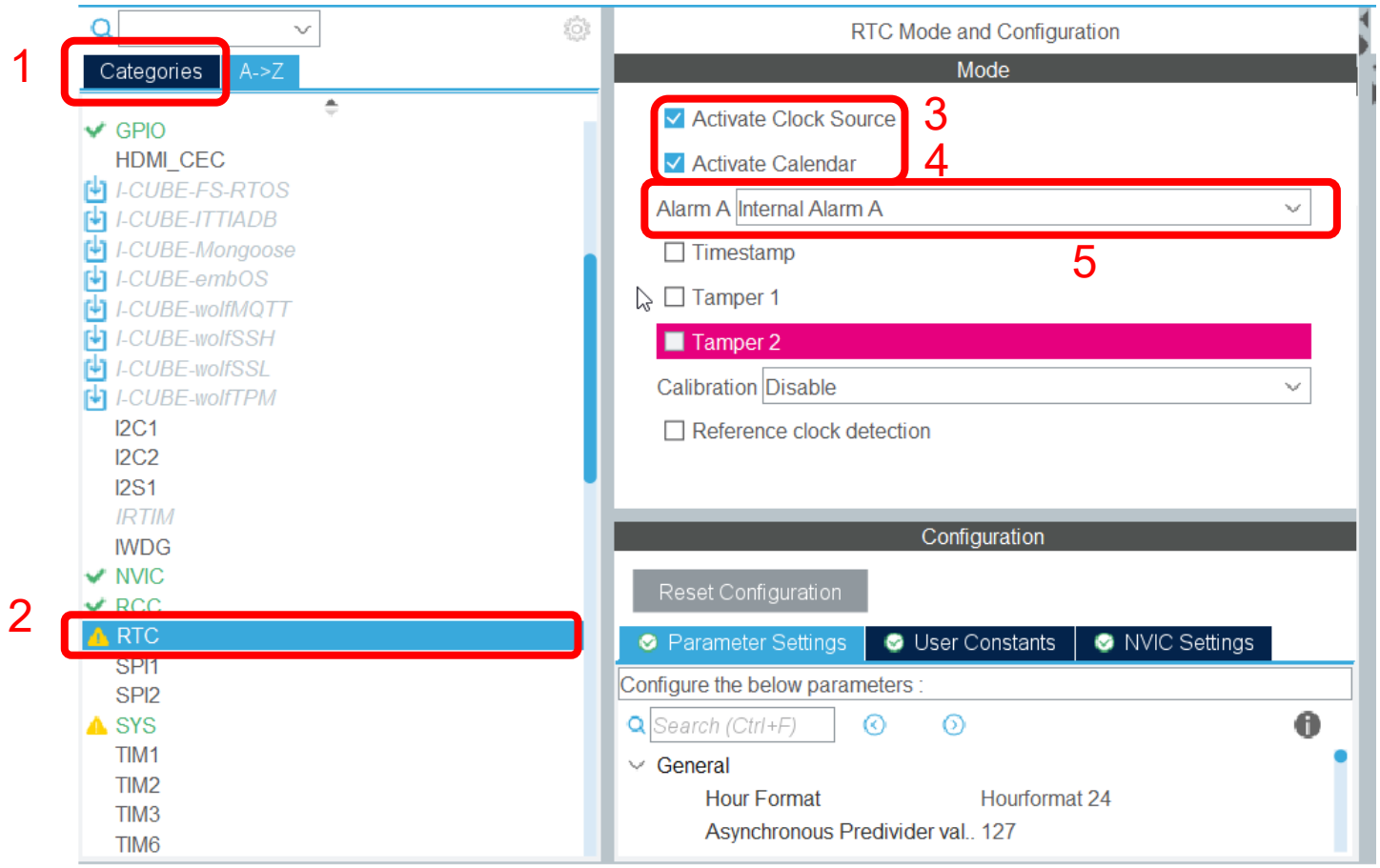
مثال کاربردی واحد RTC

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.

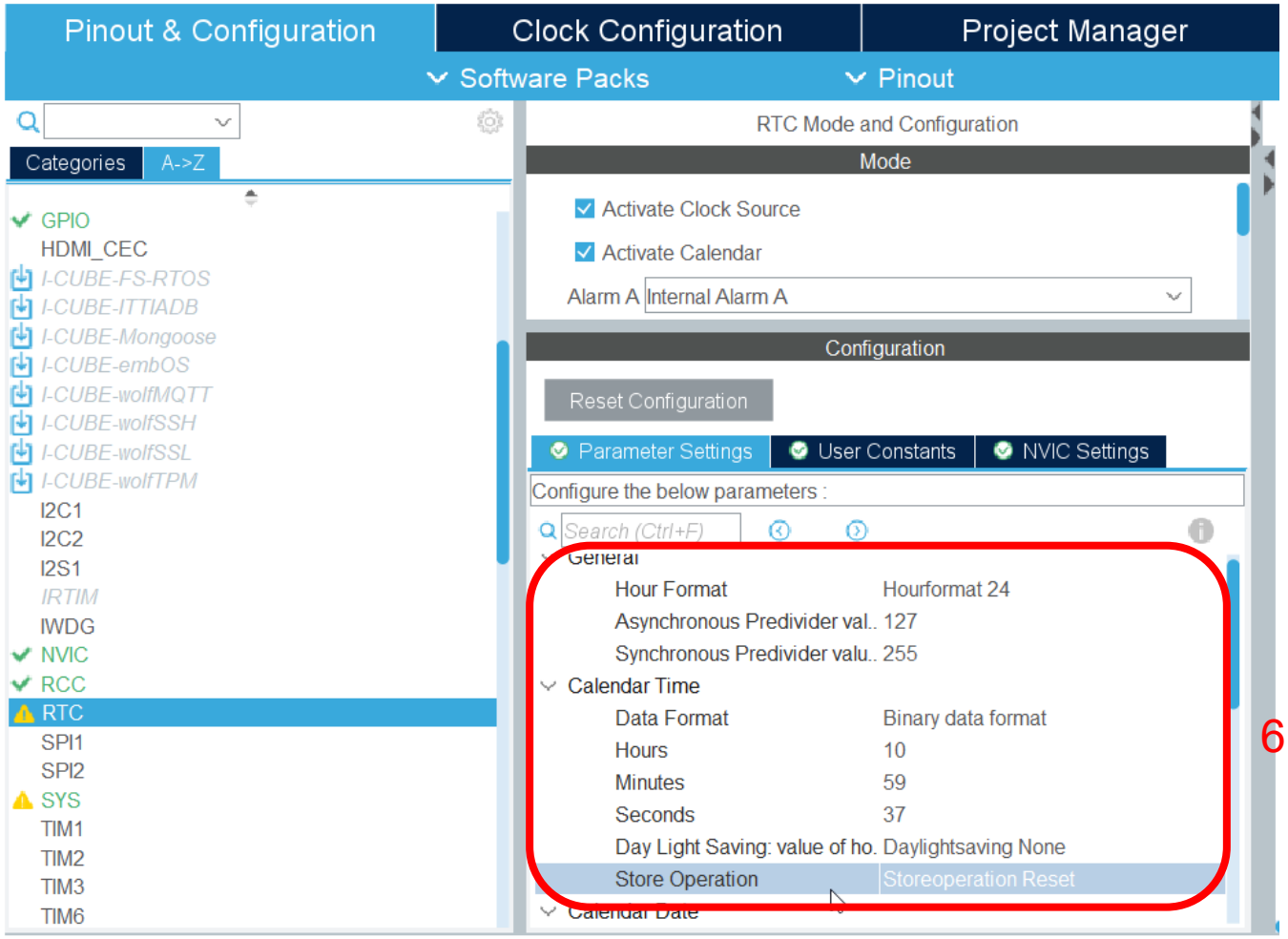
```
/* USER CODE BEGIN WHILE */
lcd_init();
lcd_gotoxy(6,0);
lcd_hide_cursor();
lcd_puts("IUST");
lcd_gotoxy(1,1);
lcd_puts("uProcessor LAB");
HAL_Delay(2000);
lcd_clear();
while (1)
{
/* USER CODE END WHILE */

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

مثال کاربردی واحد RTC

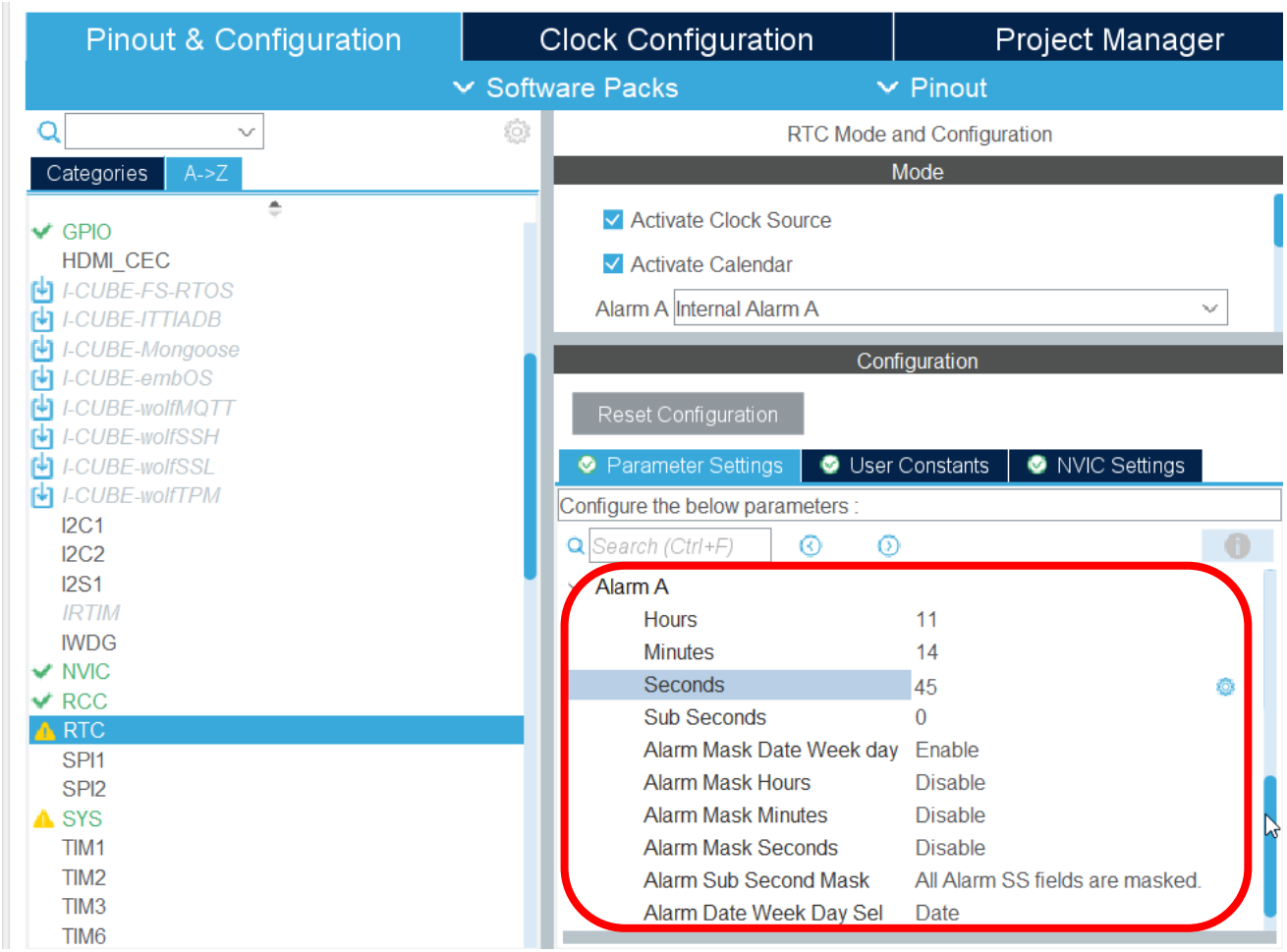
پس از اطمینان از عملکرد LCD حال برای تنظیمات و راه اندازی واحد RTC طبق تصویر زیر عمل نمایید.



مثال کاربردی واحد RTC

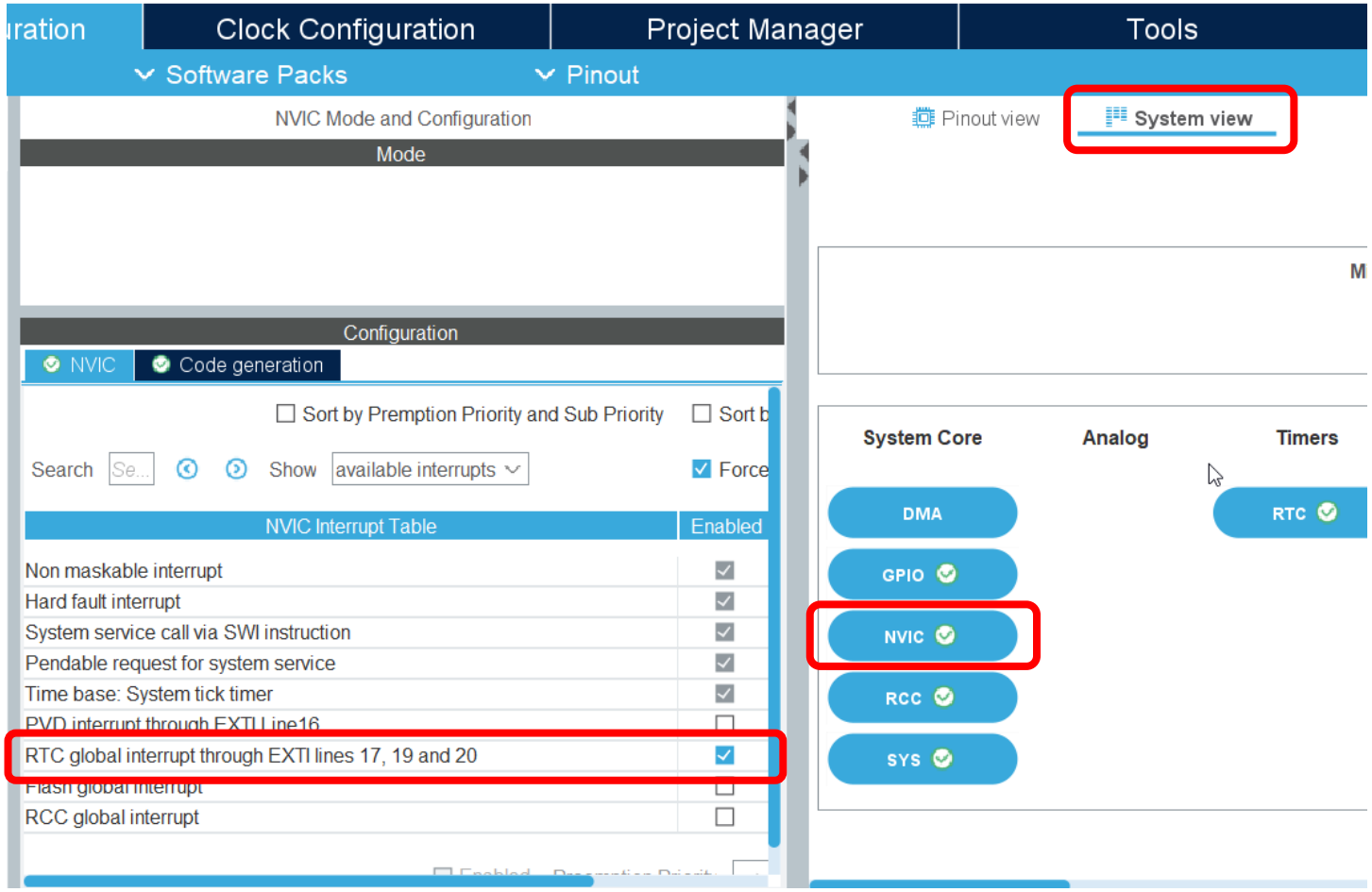


مثال کاربردی واحد RTC



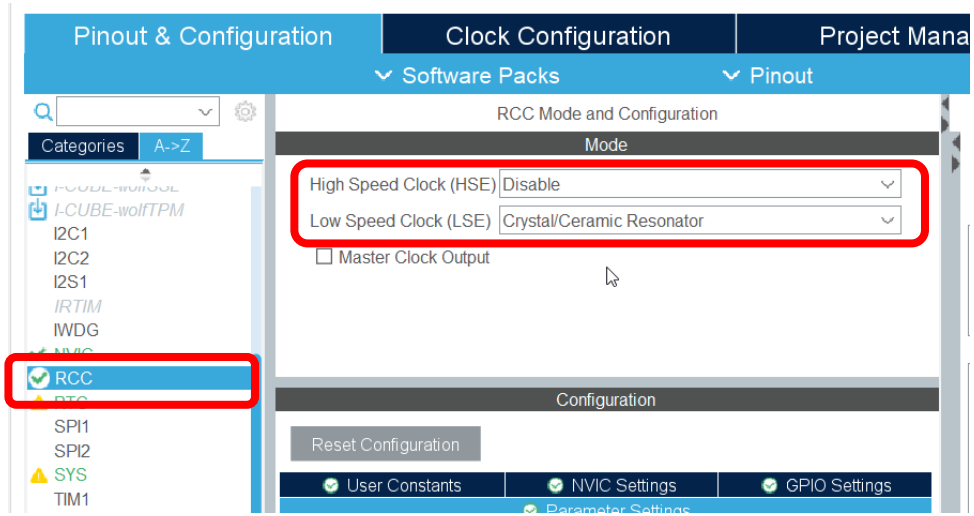
مثال کاربردی واحد RTC

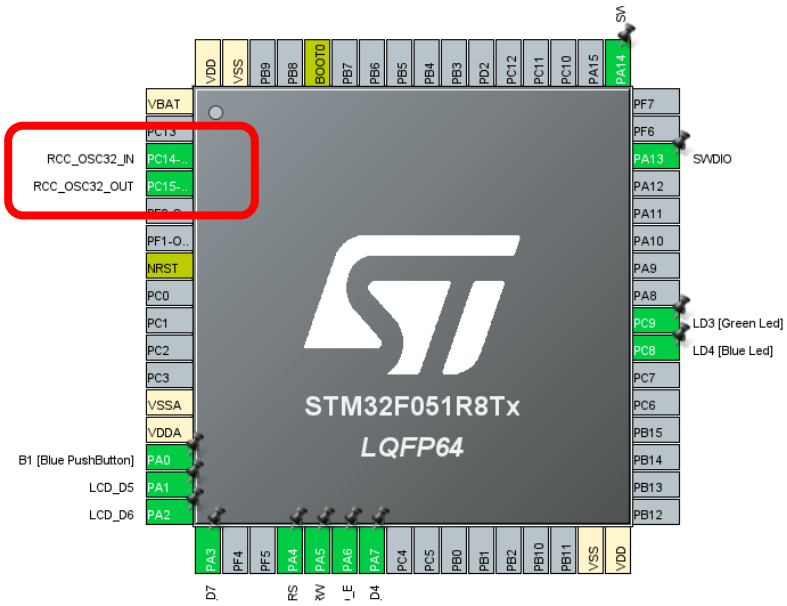
وقفه آلارم RTC را فعال کنید.



مثال کاربردی واحد RTC

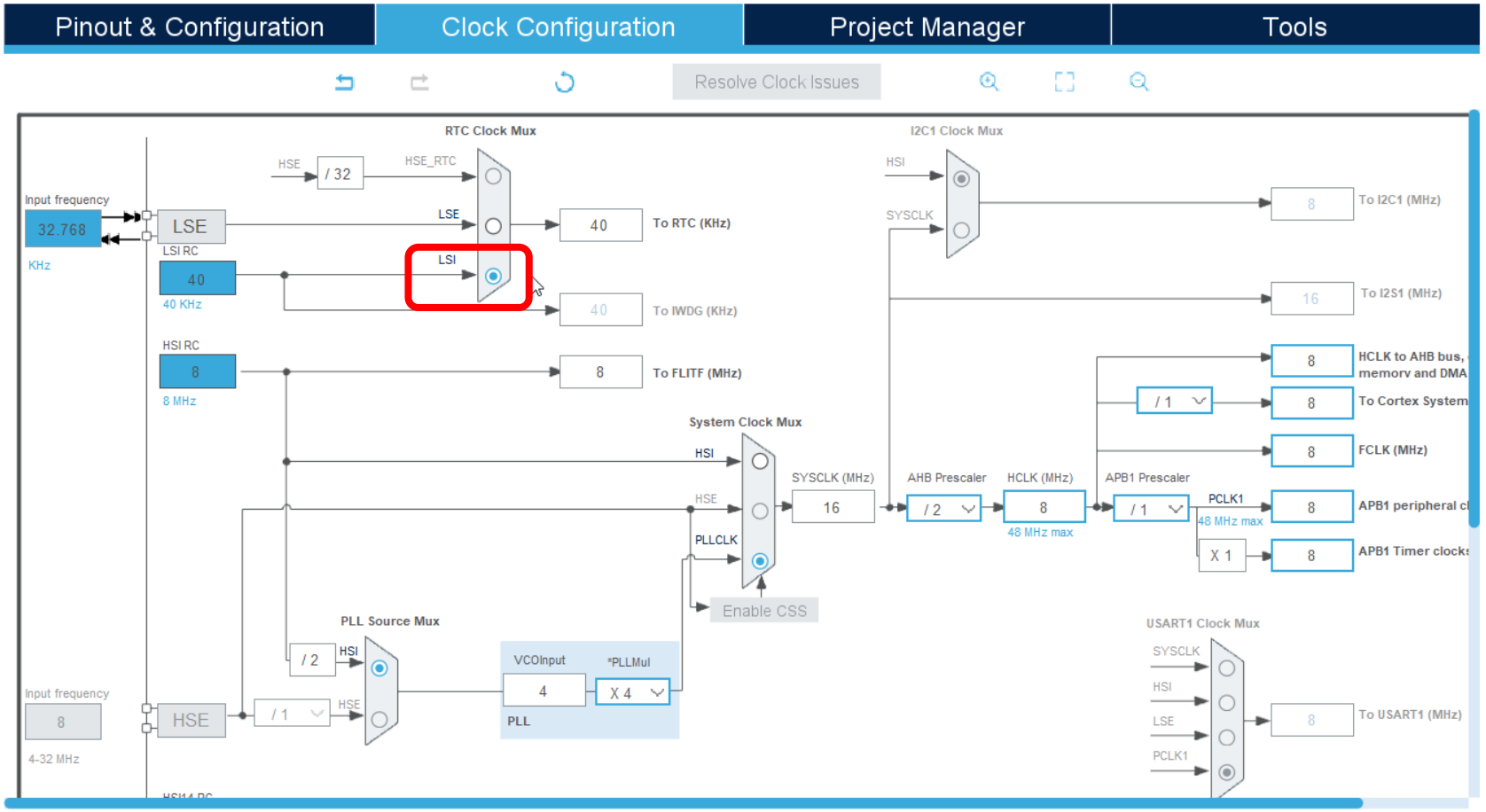
منبع کلاک مناسب را انتخاب نمایید.





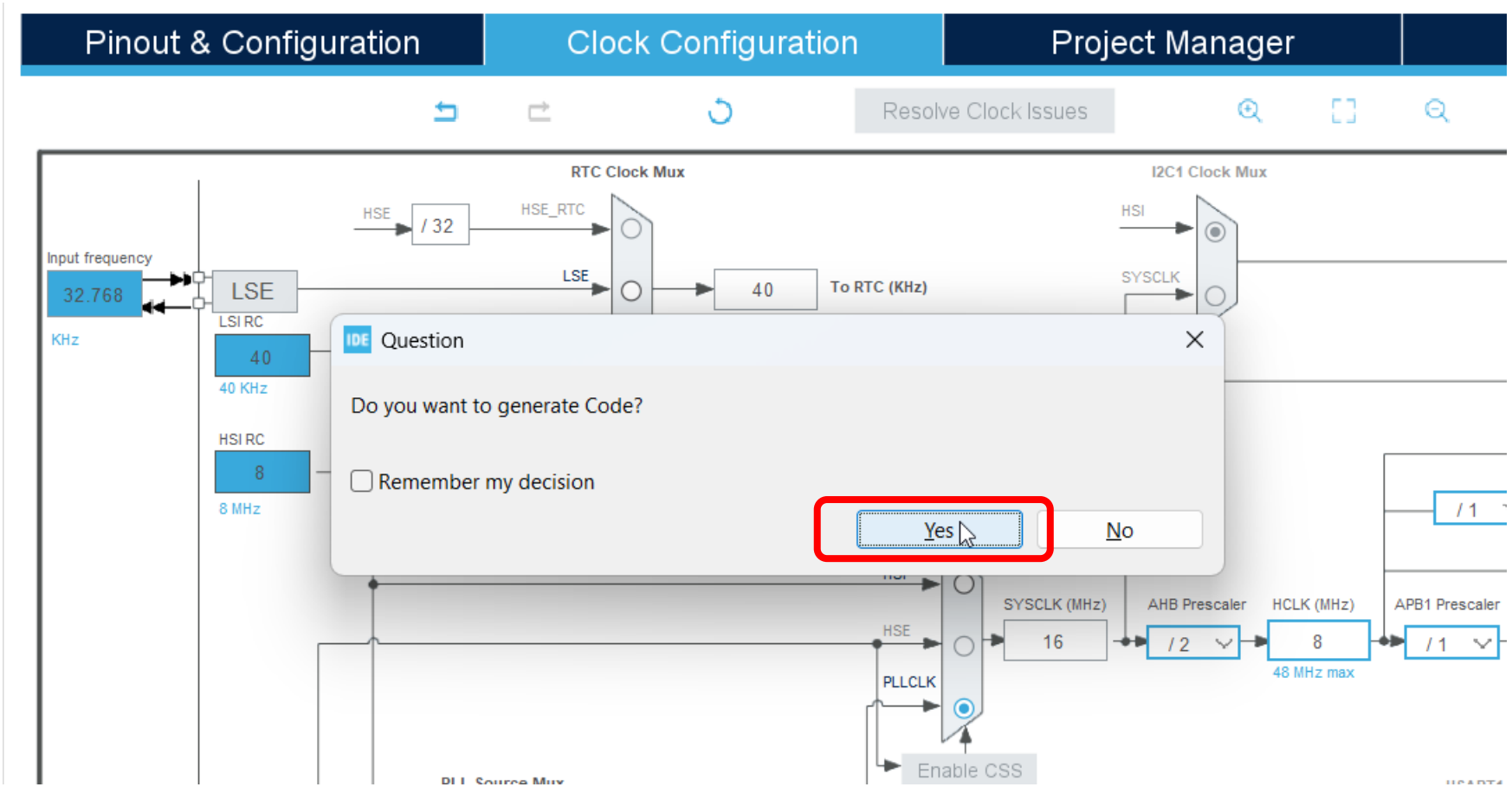
مثال کاربردی واحد RTC

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];
```

براي خواند زمان، تاريخ و آلارم به سه استراکچر نياز داريم:

```
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.Minut
    es,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.Minut
    es,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.Minut
    es,gTime.Seconds);
    lcd_gotoxy(0,0);
    lcd_puts(buffer_1);
    sprintf(buffer_1,"Alarm:%02u:%02u:%02u",gAlarm.AlarmTime.Hours,gAl
    arm.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