تایمر WATCHDOG در میکروکنترلر های AVR و STM32

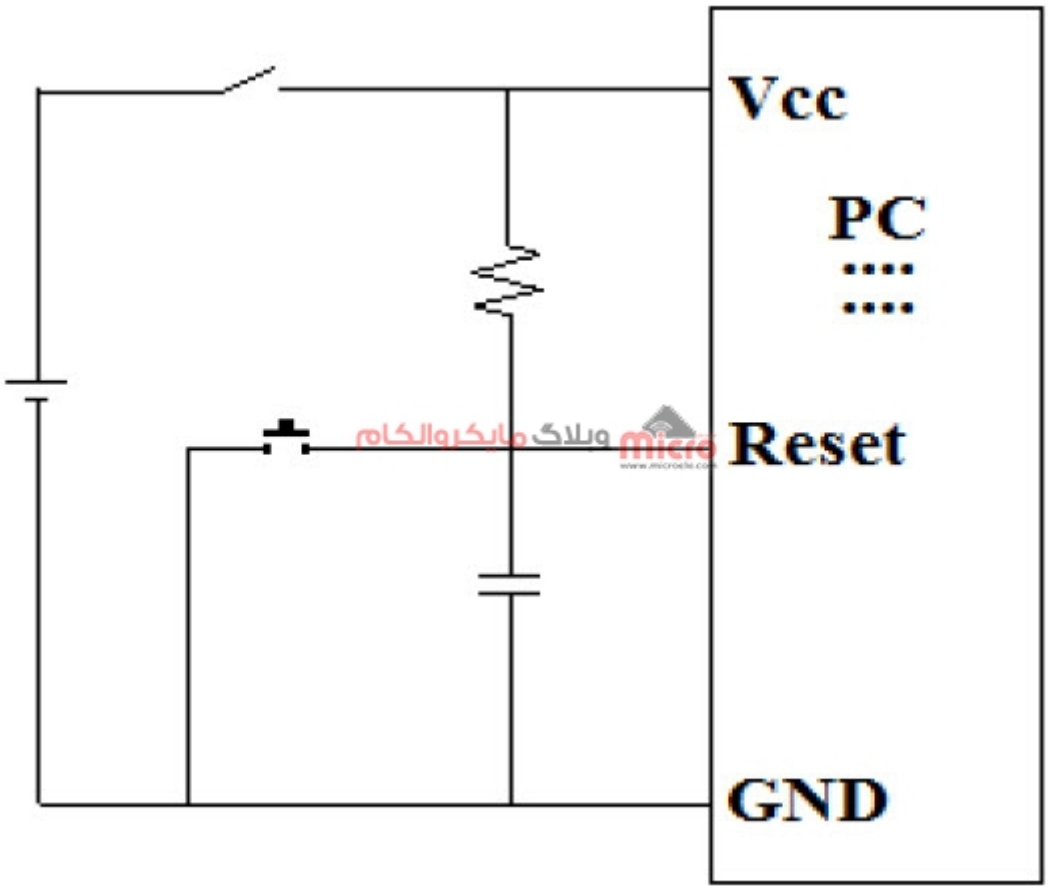
تاریخ انتشار۱۷ مرداد, ۱۴۰۱ توسط آرش فتاحی

با عرض سلام خدمت همراهان سایت مایکروالکام. در این مطلب به معرفی Watchdog در میکروکنترلرهای AVR و STM32 و نحوه راه اندازی این واحد در این میکروکنترلر ها خواهیم پرداخت. این مطلب شامل دو قسمت بوده که در قسمت اول به معرفی واحد Watchdog و مزایای استفاده از آن در میکروکنترلر و نحوه راه اندازی آن در میکرو های AVR و STM32 پرداخته شده است. پس با من تا انتهای مطلب همراه باشید. همچنین میتونید سایر مطالب من رو از این قسمت مطالعه کنید.

تایمر Watchdog و کاربرد آن

هر زمان که سیستمی اصطلاحاً هنگ کرده و موقتاً ادامه کار آن دچار اختلال گردد، لازم است تا آن سیستم دوباره از ابتدا راه اندازی شود. در کامپیوترها این عمل با فشردن کلید Reset انجام شده که با این کار سیستم عامل کامپیوتر مجدداً بوت شده و سیستم از ابتدا راه اندازی می‌شود. در میکروکنترلر ها و سیستم های نهفته (Embedded) نیز پایه‌ای برای ریست کردن وجود دارد که کاربر با فشردن آن باعث راه اندازی مجدد میکرو و شروع برنامه از ابتدا خواهد شد.

- هرگاه عمل ریست اتفاق می‌افتد، شمارنده برنامه یا PC (Program Counter) برابر با صفر شده، در نتیجه برنامه مجدداً از خط اول شروع می‌شود.



مدار پایه ریست در میکروکنترلر

از آن‌جایی که بحث ما درباره سیستم های نهفته (Embedded System) می‌باشد، ممکن است امکان پایش مدار به صورت مستمر و ریست کردن میکرو در صورت بروز مشکل توسط کاربر وجود نداشته و یا کاربر مستقیما به کلید ریست میکروکنترلر دسترسی نداشته باشد. در این جا تایمر واچ داگ (Watchdog) به کمک طراحان سخت افزار آمده و در صورتی که میکرو هنگ کند، این تایمر، میکرو را بعد از زمان مشخصی به صورت خودکار ریست کرده و میکروکنترلر از ابتدا شروع به کار خواهد کرد.

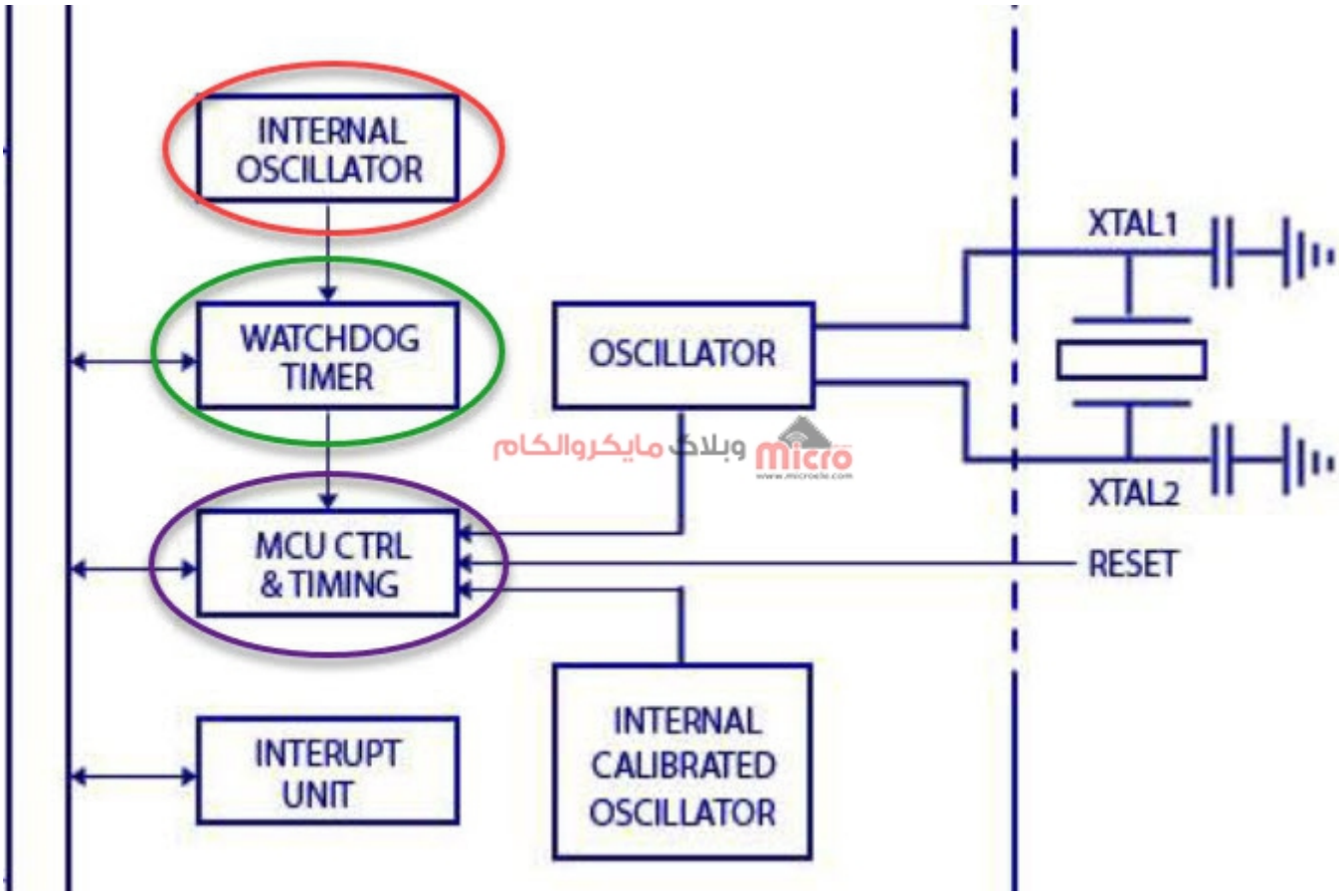
عملکرد تایمر Watchdog

عملکرد واچ داگ به این صورت است که اگر این واحد توسط میکرو ریست نشود، واچ داگ در مقابل، میکرو را ریست خواهد کرد. بنابراین اگر میکرو هنگ کرده و نتواند واچ داگ را ریست کند، این واحد به صورت مستقل عمل

کرده و اقدام به ریست کردن میکرو می‌کند. استفاده از واچ داگ در انجام پروژه ها بسیار مهم بوده تا در هنگام بروز مشکل در برنامه و هنگ کردن سیستم، امکان ریست شدن مدار به صورت خودکار وجود داشته باشد.

واچ داگ (Watchdog) در میکروکنترلرهای AVR

ابتدا برای درک بهتر Watchdog، به توضیح و راه‌اندازی این واحد در میکروکنترلرهای AVR خواهیم پرداخت. تصویر زیر بخشی از بلوک دیاگرام داخلی میکروکنترلر ATMEGA32 که مربوط به بخش واچ داگ (Watchdog) در این میکروکنترلر است را نشان می‌دهد.



بلوک دیاگرام میکروکنترلر *ATMEGA32* و واحد *Watchdog*

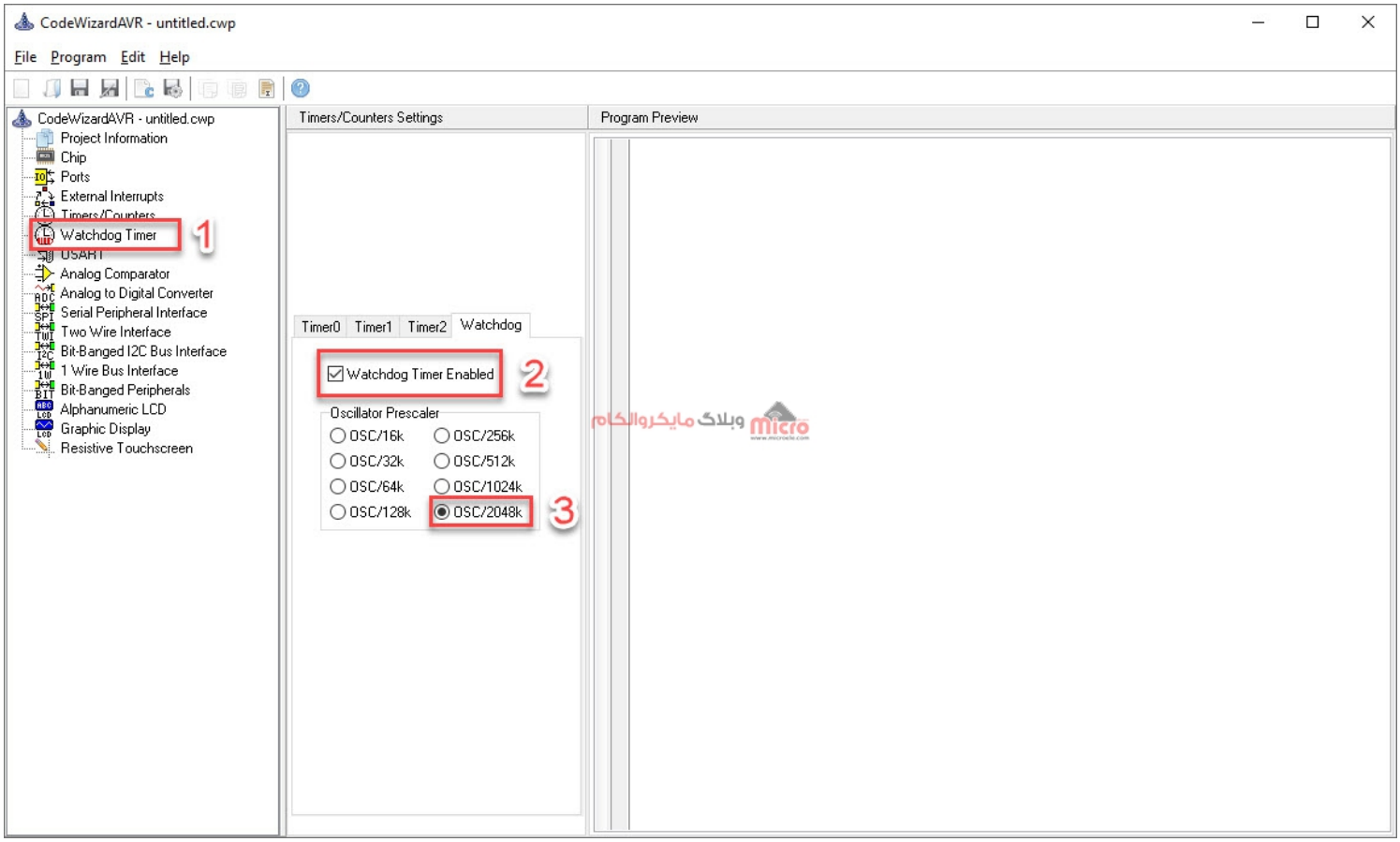
قبل از کار با این واحد در میکروکنترلرهای AVR، لازم است تا فرکانس کاری این واحد از دیتاشیت مربوط به میکرو

استخراج شود. طبق بلوک دیاگرام بالا، Watchdog Timer در این میکروکنترلر به یک اسیلاتور داخلی متصل بوده که بر اساس دیتاشیت میکرو، فرکانس آن **1 مگاهرتز** می‌باشد.

راه اندازی واچ داگ در محیط Codevision

- برای راه‌اندازی واچ داگ ابتدا Wizard را در کدویژن اجرا کرده، میکروکنترلر مدنظر خود در قسمت Chip که در این جا برای ما ATMEGA32 می‌باشد را انتخاب و به بخش Watchdog در کد ویزارد می‌رویم.
- در بخش بعدی تیک گزینه Watchdog Timer Enabled را زده تا واچ داگ فعال گردد. با فعال شدن واچ داگ، در محیط ویزارد مقسم‌های فرکانس این واحد برای اعمال تنظیمات نمایش داده می‌شود.

به عنوان مثال ما در این جا گزینه OSC/2048k را انتخاب کرده‌ایم که این به معنی تقسیم فرکانس داخلی این واحد که برابر با 1 مگاهرتز می‌باشد بر مقدار 2097152 است **(k=1024)**.



تنظیم *Watchdog timer* در محیط کدویزارد کدویژن

$$f = \frac{1000000}{2048 \times 1024} = 0.476837158203125Hz ==> \quad T = \frac{1}{f} = 2.097152s$$

طبق محاسبات بالا، زمان عملکرد واحد واچ داگ میکرو AVR به دست آمده که در اینجا مقدار 2.097 ثانیه خواهد بود. این مقدار نشان می‌دهد که اگر میکروکنترلر هنگ کند پس از سپری شدن مدت زمان 2.097 ثانیه واچ داگ به صورت خودکار اقدام به ریست کردن میکرو خواهد کرد.

ریست کردن Watchdog در میکروکنترلر AVR

برای ریست کردن واحد واچ داگ در میکروکنترلرهای AVR کافیست از دستور اسمبلی WDR که مخفف Watchdog Reset است استفاده کنیم. برای نوشتن دستورات اسمبلی در محیط کدویژن و در برنامه نوشته شده به زبان C در این کامپایلر از #asm استفاده می‌کنیم.

```
#asm("WDR")
```

مثال: آزمایش Watchdog در میکروکنترلر AVR

به عنوان نمونه در برنامه نوشته شده، یک شمارنده باینری بر روی PORTA میکروکنترلر ایجاد شده که با هر بار اضافه شدن به مقدار متغیر i، واحد واچ داگ یک بار ریست خواهد شد. در صورتی که میکرو هنگ کند دستور WDR اجرا نشده و میکروکنترلر بعد از زمان تعیین شده، توسط واحد Watchdog ریست می‌شود و شمارش بر روی LED ها از ابتدا آغاز خواهد شد.

نکته: توجه داشته باشید که برای تست برنامه نوشته شده، بهتر است از توابع delay کدویژن استفاده نکنید. زیرا امکان ریست شدن واحد Watchdog از داخل توابع delay پیشفرض در کدویژن وجود دارد.

برای جلوگیری از ریست شدن واچ داگ از طریق توابع delay در کدویژن، در برنامه نوشته شده برای ایجاد تاخیر، یک تابع به نام my_delay ایجاد کرده‌ایم که با زیاد کردن عدد مقدار شرط for در این تابع و بیشتر کردن تاخیر، می‌توانید ریست شدن میکرو به دلیل عدم رسیدن به موقع به دستور WDR را مشاهده کنید.

```
#include <mega32.h>

unsigned char i=0;

void my_delay(void);

void main(void)
{
// Watchdog Timer initialization
// Watchdog Timer Prescaler: OSC/2048k
WDTCR=(0<<WDTOE) | (1<<WDE) | (1<<WDP2) | (1<<WDP1) | (1<<WDP0);

while (1)
          {
          // Place your code here
          PORTA=i;
          i++;
          #asm("WDR")
          my_delay();
          }
}

void my_delay(void)
{
 unsigned int i;
 for(i=0;i<5000;i++);
}
```

```
// TWI initialization
// TWI disabled
TWCR=(0<<TWEA) | (0<<TWSTA) | (0<<TWSTO) | (0<<TWEN) | (0<<TWIE);

// Watchdog Timer initialization
// Watchdog Timer Prescaler: OSC/2048k
WDTCR=(0<<WDTOE) | (1<<WDE) | (1<<WDP2) | (1<<WDP1) | (1<<WDP0);

while (1)
        {
        // Place your code here
        PORTA=i;
        i++;
        #asm("WDR")
        my_delay();

        }
}

void my_delay(void)
{
 unsigned int i;
 for(i=0;i<5000;i++);
}
```

دستور ریست کردن *Watchdog* در کدویژن

واچ داگ در میکروکنترلرهای STM32

میکروکنترلرهای STM32 دارای دو نوع واچ داگ به نام‌های واچ داگ مستقل (IWDG) و واچ داگ پنجره (WWDG) می‌باشد. در این بخش به معرفی این دو حالت خواهیم پرداخت و همچنین در قسمت اول این مطلب خواهیم دید که

چگونه می‌توان در میکروکنترلرهای STM32 از واچ داگ‌های IWDG (Independent Watchdog) یا واچ داگ مستقل استفاده کرد.

هدف استفاده از هر دو حالت Watchdog در میکروکنترلرهای STM32 ریست شدن میکرو در هنگام بروز مشکل در اجرای نرم افزار می‌باشد. اما تفاوت آن‌ها در نحوه پیاده سازی این دو است. بزرگ‌ترین تفاوت میان این دو در این است که می‌توان واچ داگ مستقل (IWDG) را هر زمان، قبل از این که اتمام زمان آن اتفاق بی‌افتد ریست کرد. اما واچ داگ پنجره (WWDG) تنها در بخش‌هایی از پنجره زمانی ریست می‌گردد که در قسمت دوم آن را شرح خواهیم داد.

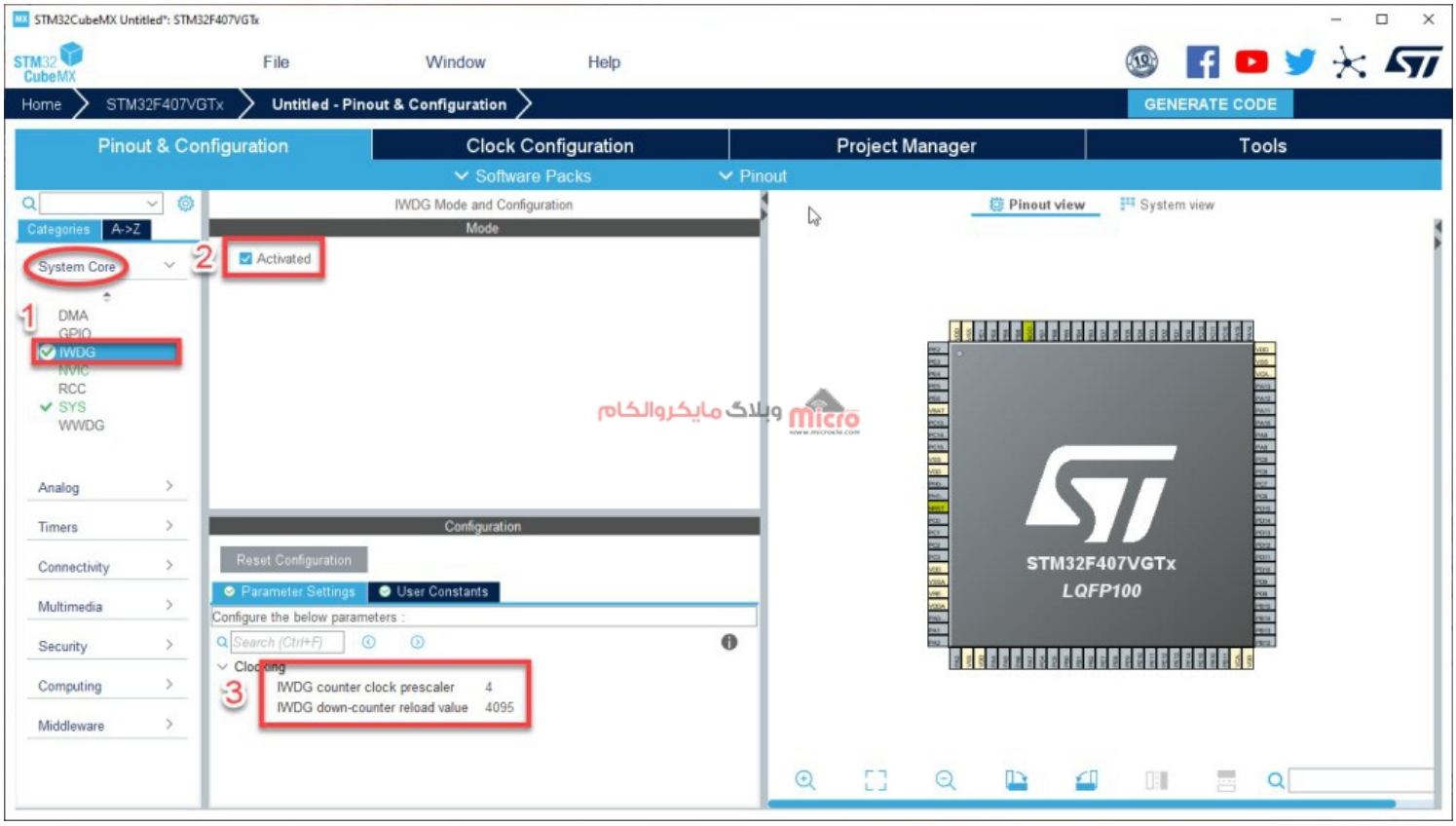
واچ داگ مستقل یا IWDG در STM32

واچ داگ مستقل برای زمان‌هایی که برنامه دچار مشکل شده و عملاً ادامه اجرای برنامه دچار اختلال می‌شود استفاده می‌گردد. هنگامی که در وقت تعیین شده، عملیات رفرش واچ داگ صورت نپذیرد، واچ داگ اقدام به ریست کردن میکرو خواهد کرد. از آن‌جایی که کلاک این واحد، مستقل از کلاک اصلی سیستم و از اسیلاتور داخلی 32 کیلو هرتز RC کم سرعت (LSI) تامین می‌گردد، این بخش همواره به صورت فعال باقی خواهد ماند حتی اگر منبع اصلی کلاک میکرو قطع شده و یا دچار اختلال شود.

زمانی که این واحد فعال گردد، اسیلاتور کم سرعت داخلی میکرو همراه با آن فعال شده و فقط با به روز رسانی کردن IWDG می‌توان از ریست شدن میکرو توسط واچ داگ جلوگیری کرد. عملکرد به‌روزرسانی IWDG همچون دستور WDR در میکروکنترلر های AVR می‌باشد.

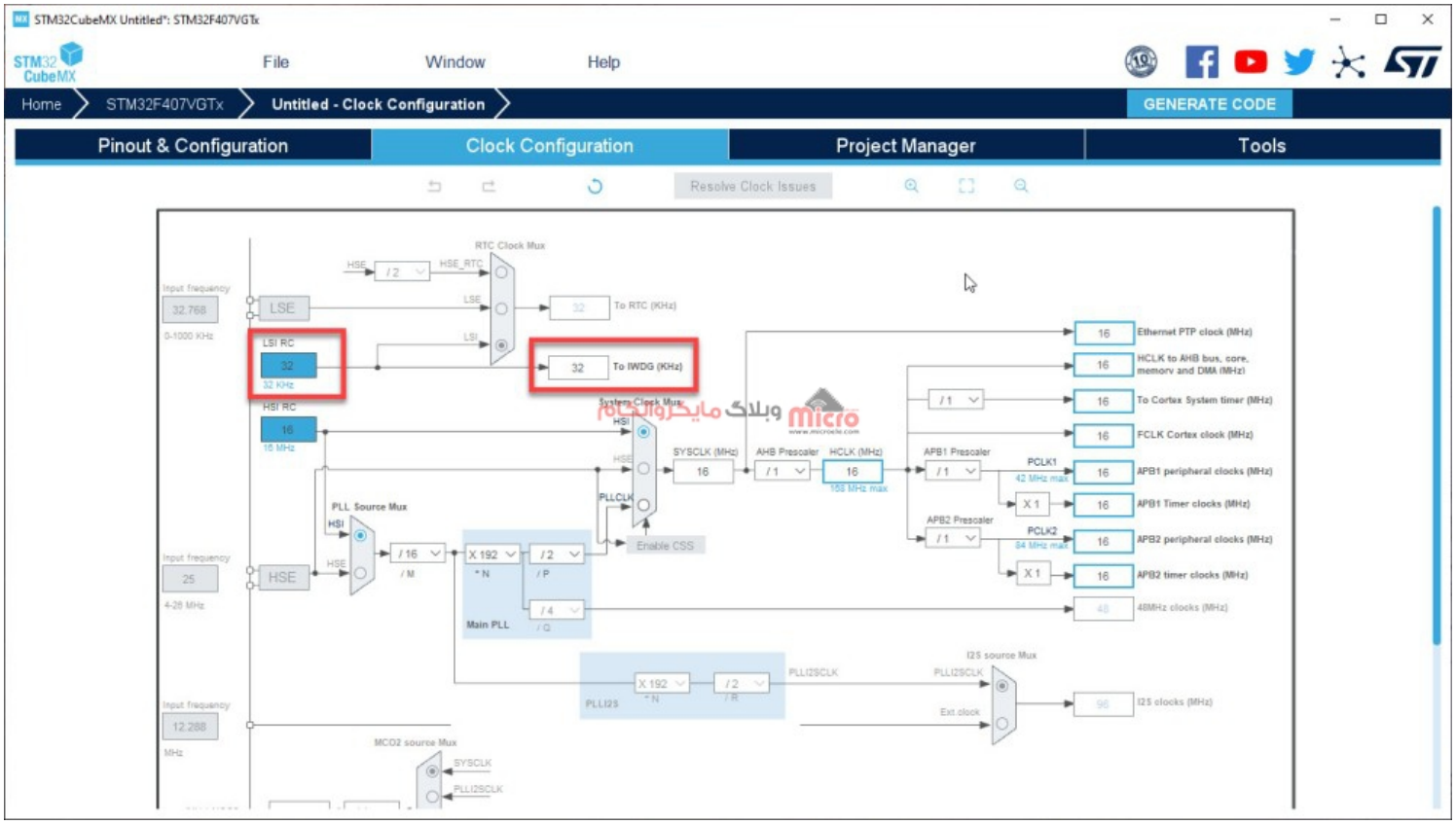
تنظیمات محیط CubeMX برای IWDG

بعد از اجرای نرم افزار CubeMX و انتخاب میکرو (در این جا STM32F407VGT) به بخش System Core رفته و IWDG را انتخاب کنید. در مرحله بعد تیک گزینه Activated را زده و واچ داگ را فعال کنید.



تنظیمات *Watchdog* در محیط *CubeMX*

همچنین با باز کردن سربرگ Clock Configuration در CubeMX می‌توانید فعال شدن کلاک این واحد که برابر با 32Khz می‌باشد را مشاهده کنید.



فرکانس واحد *IWDG* در *Clock Configuration*

فرمول محاسبه Timeout واچ داگ در میکروکنترلرهای STM32

همانطور که در تصویر مربوط به تنظیمات واحد IWDG (تصویر قبل) مشاهده کردید، در بخش Parameters Settings دو پارامتر Prescaler و مقدار Counter وجود دارد که باید با توجه به نیاز برنامه نویس محاسبه و تنظیم گردد. این مقادیر، زمان لازم جهت ریست کردن میکرو توسط واچ داگ را مشخص می‌کنند. برای انتخاب و محاسبه این مقادیر می‌توان از فرمولی که توسط ST ارائه شده است بهره برد.

فرمول محاسبه *Timeout* در میکروکنترلرهای *STM32*

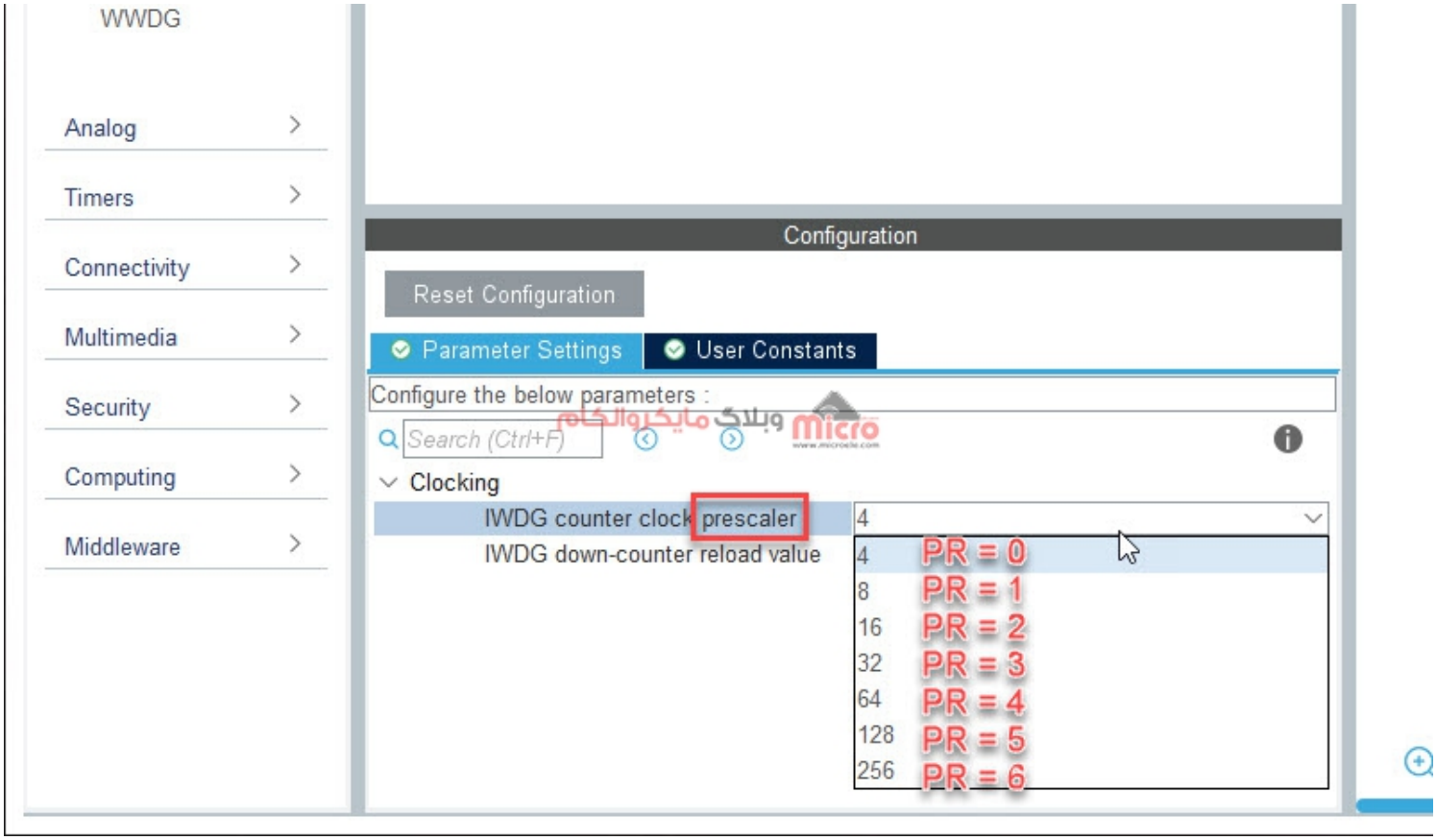
$$RL=\frac{Time_{(ms)}\times 32000}{4\times 2^{(PR)}\times 1000}-1$$

فرمول محاسبه *Timeout* واچ داگ در میکروکنترلرهای *STM32*

توضیحات فرمول

در این بخش به توضیح فرمول ارائه شده در بالا خواهیم پرداخت:

- RL مقدار شمارنده (Counter) می‌باشد که باید در تنظیمات قرار داده شود.
- Time مقدار زمانی است که باید سپری شود تا میکرو توسط Watchdog ریست گردد. این مقدار به میلی ثانیه است. اگر کانتر قبل از این مدت به روز رسانی نگردد میکرو ریست خواهد شد.
- PR مقدار Prescaler بوده که مقادیر آن در تصویر زیر قابل مشاهده است.



تنظیمات *IWDG* در محیط *CubeMX*

حل یک مثال و محاسبه Timeout

به عنوان مثال Watchdog را با استفاده از فرمول گفته شده به گونه ای تنظیم می‌کنیم که در صورت ریست نشدن واچ داگ بعد از 20 میلی ثانیه، میکروکنترلر ریست گردد.

- در این‌جا به دلخواه Prescaler را بر روی 8 تنظیم کرده‌ایم. بنابراین PR با توجه به شکل بالا برابر با 1 خواهد شد (PR=1).
- در فرمول Time، IWDG را نیز 20 میلی ثانیه قرار خواهیم داد.
- با قرار دادن مقادیر گفته شده در فرمول، مقدار RL برابر با 79 خواهد شد بنابراین باید مقدار Counter Value را برابر با RL یعنی 79 در CubeMX قرار دهیم.

نکته: توجه داشته باشید در صورتی که در محاسبات انجام شده، مقدار RL از عدد 4095 بیشتر گردد، باید مقدار Prescaler را روی عدد بالاتری تنظیم کنید.

قبل از تولید کد در محیط CubeMX، یکی از پایه‌های میکرو را به عنوان خروجی (GPIO_Output) تعریف کرده که در نهایت به این پایه یک LED برای انجام آزمایش متصل خواهیم کرد (در این‌جا ما پایه PD12 را به عنوان خروجی تعریف کرده‌ایم). بعد از اعمال تنظیمات بر روی Generate code کلیک کرده تا پروژه تولید گردد.

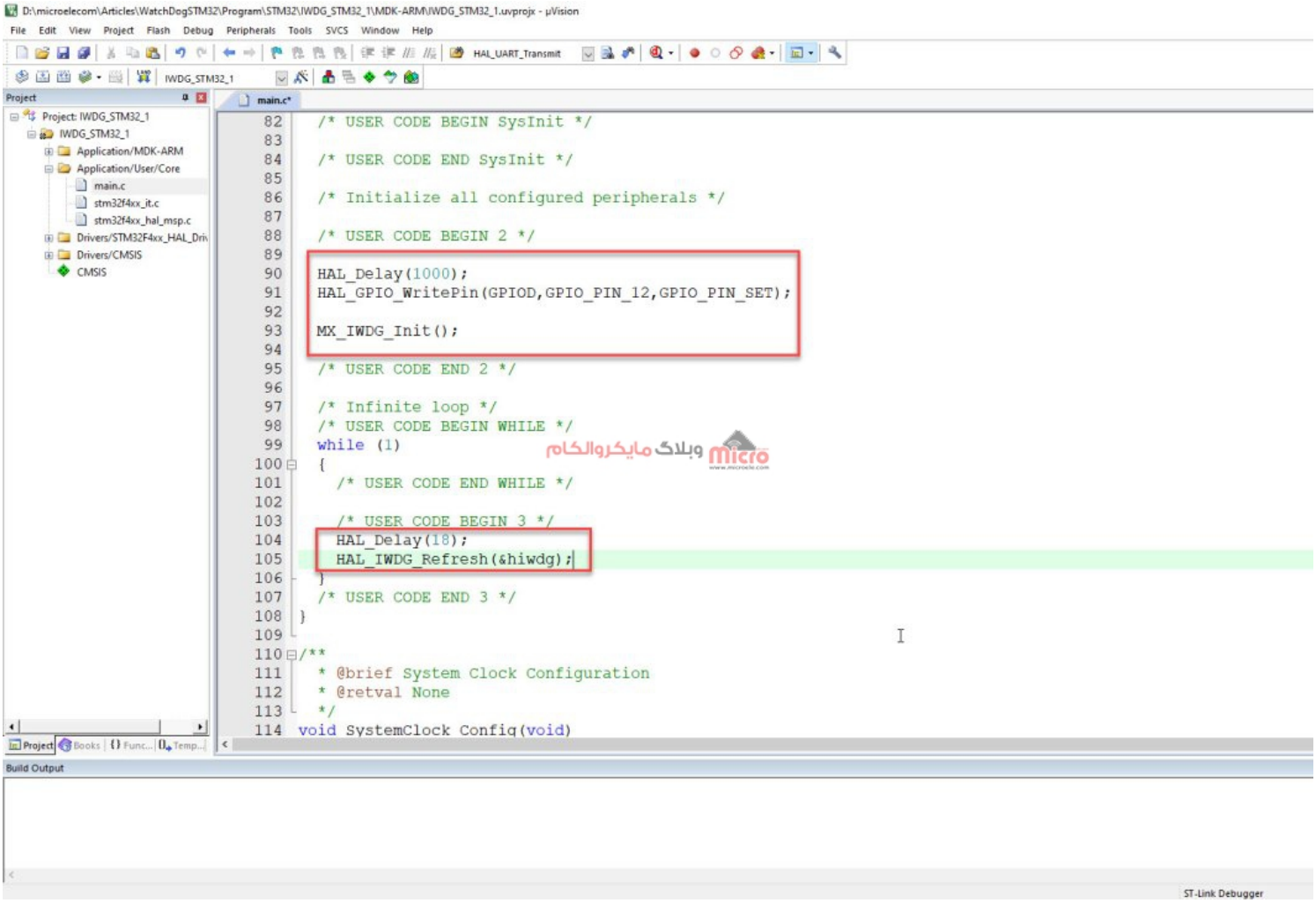
شروع برنامه نویسی در محیط Keil برای تست Watchdog در STM32

در این بخش یک برنامه ساده برای تست عملکرد IWDG خواهیم نوشت. برای تست واچ داگ قبل از حلقه While یک LED را بعد از تاخیر 1 ثانیه‌ای روشن خواهیم کرد. در صورتی که در حلقه while، واچ داگ را ریست نکنیم واچ داگ میکروکنترلر را ریست خواهد کرد و LED شروع به چشمک زدن خواهد کرد.

برای initialize کردن و شروع به کار واحد IWDG از تابع MX_IWDG_Init استفاده می‌شود که توسط خود CubeMX در کد به صورت پیش فرض نوشته شده است. قبل از این دستور، 1000 میلی ثانیه صبر کرده و سپس LED را روشن می‌کنیم. دلیل این که قبل از دستور MX_IWDG_Init این کار را انجام دادیم این است که تا قبل از روشن شدن LED واچ داگ فعال نشده و میکرو ریست نگردد.

در حلقه While بعد از 18 میلی ثانیه تاخیر، با استفاده از تابع HAL_IWDG_Refresh(&hiwdg) اقدام به ریست کردن واحد IWDG کرده و از ریست شدن میکرو جلوگیری می‌کنیم.

در صورتی که دستور IWDG_Refresh را حذف کنیم، میکرو دائما ریست شده و شاهد چشمک زدن LED خواهیم بود اما با باقی ماندن آن در حلقه While، میکرو دیگر ریست نشده و LED به صورت ثابت روشن خواهد ماند.



کدنویسی برای بررسی *Watchdog* در محیط *Keil* برای *STM32*

```
HAL_Delay(1000);
    HAL_GPIO_WritePin(GPIOD,GPIO_PIN_12,GPIO_PIN_SET);
    MX_IWDG_Init();

  /* USER CODE END 2 */

  /* Infinite loop */
```

```
/* USER CODE BEGIN WHILE */
while (1)
{
  /* USER CODE END WHILE */

  /* USER CODE BEGIN 3 */
      HAL_Delay(18);
      HAL_IWDG_Refresh(&hiwdg);
}
/* USER CODE END 3 */
```

نتیجه گیری

در این مطلب به توضیح واچ داگ (Wathcdog) و مزایای استفاده از آن پرداخته شد. همچنین نحوه استفاده و راه اندازی این واحد در میکروکنترلرهای AVR با کامپایلر کدویژن و میکروکنترلرهای STM32 توضیح داده شد. در آینده و در قسمت دوم، به سراغ WWDG و نحوه استفاده از آن در میکرو های STM32 خواهیم رفت.

امیدوارم از این آموزش کمال بهره را برده باشید. در صورتی که هرگونه نظر یا سوال داشتید درباره این آموزش لطفا اون رو در انتهای همین صفحه در قسمت دیدگاه ها قرار بدید. در کوتاه ترین زمان ممکن به اون ها پاسخ خواهم داد. اگر این مطلب براتون مفید بود، اون رو حتما به اشتراک بگذارید. همینطور میتونید این آموزش را پس از اجرای عملی توی اینستاگرام با هشتگ microelecom# به اشتراک بگذارید و پیج مایکروالکام (@microelecom) رو هم منشن کنید.