بسمه تعالی

گزارش آزمایش ۸ آزمایشگاه ریزپردازنده

سید محمدرضا حسینی

حسین حاجی رومنان

۱.

Input capture: با استفاده از قابلیت Input capture میکروکنترلر فرکانس سیگنال ورودی را میتوانیم اندازه بگیریم. برای اندازه‌گیری سیگنال ورودی ابتدا باید شمارنده‌ی واحد تایمر را با فرکانس مشخص، به‌صورت بالا شمار (یا پایین شمار) راه‌اندازی کنیم. سپس سیگنالی که می‌خواهیم فرکانس آن را اندازه‌گیری کنیم را به یکی از کانال‌های تایمر که بر روی پین میکروکنترلر قرار دارد، متصل می‌کنیم و وقفه‌ی این کانال را فعال و حساس به لبه‌ی بالارونده (یا پایین‌رونده) قرار می‌دهیم. در این صورت در هر لبه‌ی بالارونده، یک وقفه رخ می‌دهد پس‌ازاینکه وقفه‌ی مربوط به لبه‌ی بالارونده رخ داد، مقدار شمارنده در لحظه وقوع وقفه، در رجیستر Capture ذخیره می‌شود. شمارنده همچنان به شمارش خود ادامه خواهد داد. در حالی که شمارنده به شمارش خود ادامه می‌دهد، بر روی دومین لبه‌ی بالارونده سیگنال یک وقفه‌ی دیگر رخ می‌دهد پس‌ازاینکه وقفه‌ی مربوط به دومین لبه‌ی بالارونده رخ داد، مقدار شمارنده در لحظه وقوع وقفه، دوباره در رجیستر Capture ذخیره می‌شود اکنون ما یک سری اطلاعات داریم که باید با استفاده از این اطلاعات، مقدار فرکانس سیگنال ورودی را محاسبه کنیم. خب ما

اطلاعات مقدار شمارنده در لحظات وقوع وقفه‌ها که در رجیستر Capture ثبت‌شده است و همچنین فرکانس کلاک شمارنده را در اختیارداریم و برای محاسبه فرکانس در زمان‌های مختلف، مراحل بالا را به‌صورت مستمر تکرار می‌کنیم.

Output compare در این مد میکروکنترلر برای کاربرد هایی از جمله ساخت pwmاستفاده میکنیم، ابتدا تایمر تنظیم میشود و شروع به شمارش میکند، مقدار متغیر مقایسه کننده نیز تنظیم میشود، هر گاه مقدار شمارنده به مقدار مقایسه رسید، یک خروجی را تغییر وضعیت میدهد، بعبارتی با ساخت یک دیوتی سایکل میتوان پالس pwm را تولید کرد.

Pwm grneration: در این مد مخصوص تولید پالس pwm است که با تنظیم متغیرها میتوان به پالس pwm رسید، از جمله متغیرهای تنظیمی تنظیم دیوتی سایکل و فرکانس تایمر، مقدار سر ریز، وقفه ها و ... است.

: One pulse mode output

از این مود برای تولید یک تک پالس در خروجی در کاربردهای مختلف استفاده میشود.

.۲

:Auto reload register

شمارنده‌ی تایمر در میکروکنترلرهای STM۳۲ با توجه به تنظیماتی که ما اعمال کردیم از ۰ تا مقدار Auto-reload register شروع به شمارش می‌کند که بیشترین مقدار این رجیستر با توجه به ۱۶ بیتی بودن آن برابر با ۶۵۵۳۵ است .پس‌ازاینکه شمارنده به مقدار Auto-reload register رسید یک سرریز رخ می‌دهد و دوباره از ۰ شروع به شمارش می‌کند.

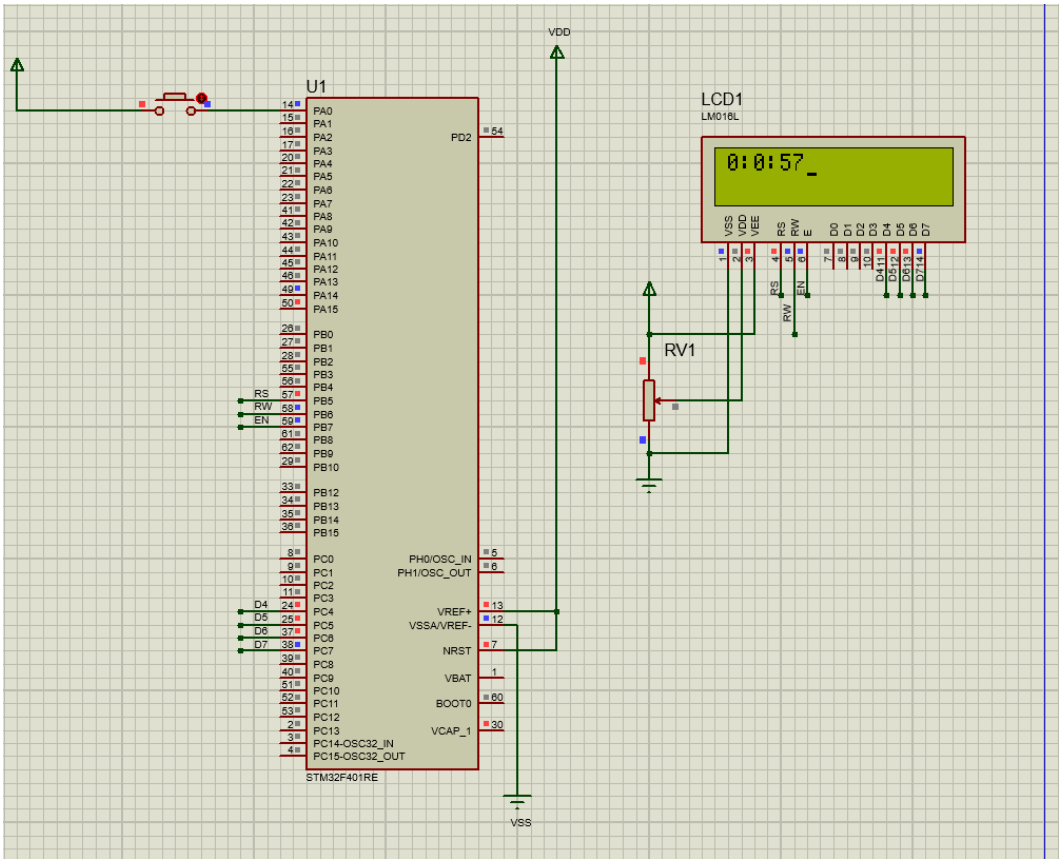
:Prescaler

Timer در میکروکنترلرهای STM۳۲ دارای یک Prescaler با طول ۱۶ بیت است که فرکانس ورودی واحد تایمر را به عددی بین ۱ تا ۶۵۵۳۶ تقسیم می‌کند. پس ما علاوه بر اینکه با استفاده از Prescalerها و ضرب‌کنندهای فرکانسی که قبل از واحد تایمر قرار دارند، می‌توانیم فرکانس ورودی واحد تایمر را تعیین کنیم، با استفاده از Prescaler که در خود واحد تایمر قرار دارد هم این انعطاف را داریم که فرکانس را تا حد بسیار زیادی، و تقریبا به هر عددی که بخواهیم تغییر بدهیم. هم Prescaler و هم Counter هر دو ۱۶ بیتی هستند و Prescaler دقیقا قبل از Counter قرار داده شده است و کلاک متصل به Counter، دقیقا همان کلاکی است که از خروجی Prescaler گرفته می‌شود.

Counter register:

این رجیستر برای ذخیره سازی مقدار تایمر و یا کاننتر در هر لحظه به کار میره و در هر لحظه قابل خواندن و نوشتنه. به ازای هر کلاک یک واحد به مقدار ایین رجیستر اضافه شده تا اینکه به مقدار ماکزیمم خود رسیده و پس از آن سریز میشه.

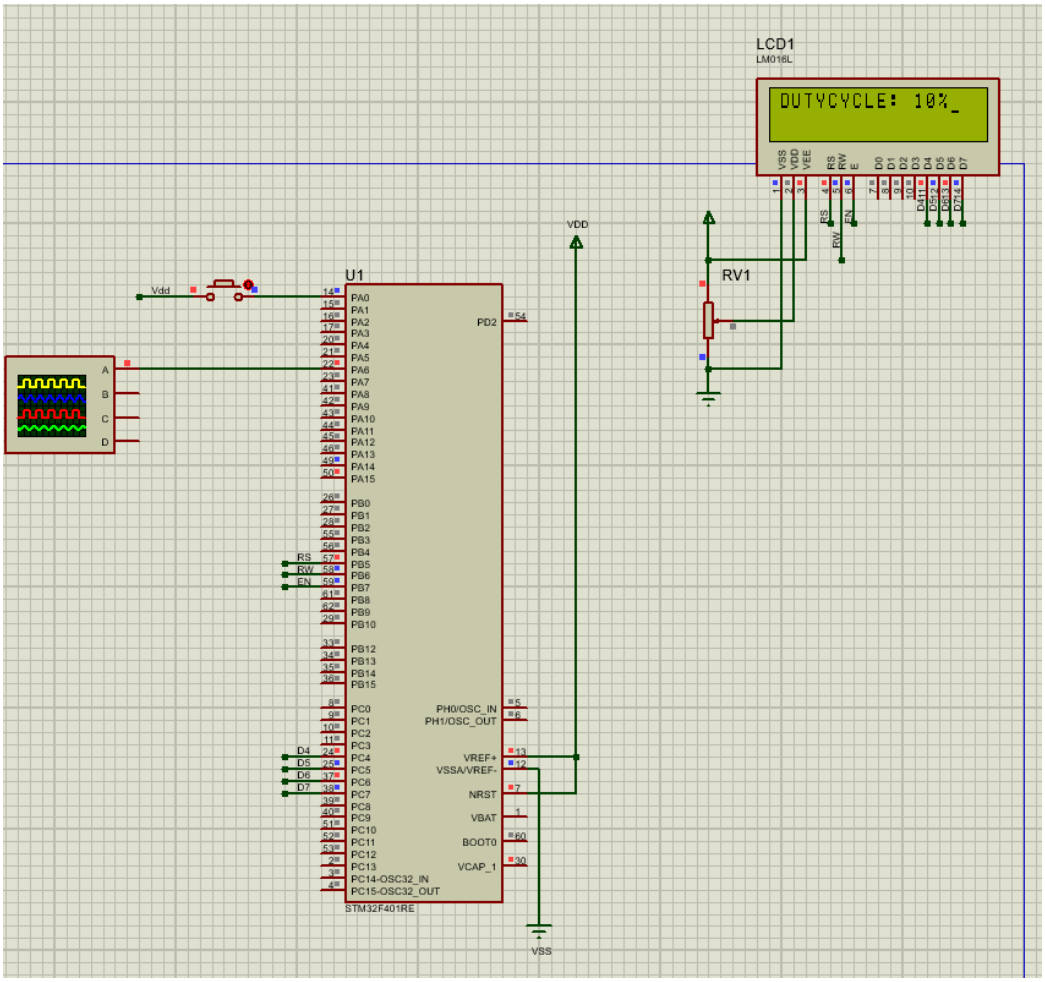
۱) سوالات پیاده سازی

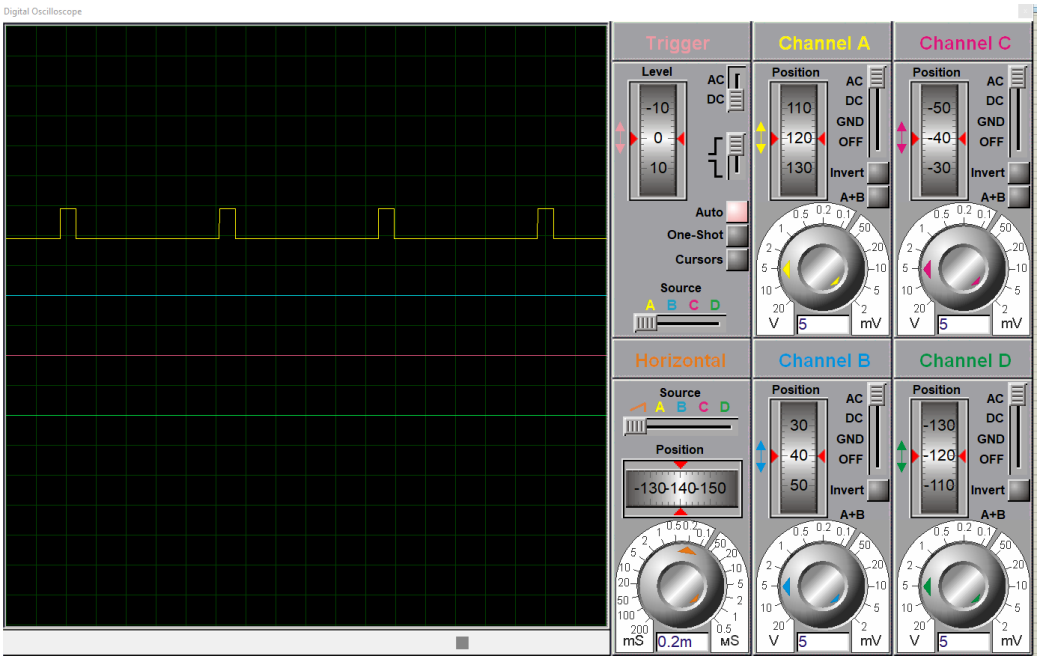


در این سوال ۳ تایمر استفاده شده است . تایمر ۲ برای Input capture استفاده شده و در در هر دو لبه کلاک مقدار کانتر را ذخیره میکند. تایمر ۳ برای نمایش اعداد بر روی LCD استفاده شده با فرکانس ۱۰۰hz . تایمر ۳ هم برای شمارش استفاده شده و فرکانس آن 1mhz میباشد.

با فشرده شدن button مقدار کانتر صفر شده و با رها کردن آن مقدار زمانی که نگه داشته شده بود اندازه گیری میشود. اگر بیش از ۵۰۰ ms باشد تایمر ۵ شمارنده ، ریست میشود. اگر این مدت کمتر از ۵۰۰ ms باشد ، در آنصورت شمارش آغاز شده یا متوقف میشود.

تایمر ۳ با استفاده از interrupt مقدار کانتر تایمر ۵ را بر روی LCD نمایش میدهد.





در شکل بالا به درستی duty cycle = ۱۰٪ مشخص است.

در این سوال تایمر ۲ برای استفاده از کلاک اکسترنال تنظیم شده به این صورت که با فشرده شدن دکمه مقدار کانتر تایمر یک واحد افزایش میابد. با توجه به اینکه ٤ دیوتی سایکل در نظر گرفتیم مقدار ARR تایمر را برابر ۳=۱-٤ قرار میدهیم . با هر بار فشرده شدن باتن ، دیوتی سایکل تغییر میکند.

```
while (1)
{
    while(count ==__HAL_TIM_GetCounter(&htim2));
    count=__HAL_TIM_GetCounter(&htim2);
    if (count==0){
    htim3.Instance->CCR1 = 1000*0.1;
    write(1);

    if (count==1){
    htim3.Instance->CCR1 = 1000*0.3;
    write(3);

    if (count==2){
    htim3.Instance->CCR1 = 1000*0.5;
    write(5);

    if (count==3){
    htim3.Instance->CCR1 = 1000*0.9;
    write(9);


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