بسمه تعالی

دانشگاه شهید باهنر کرمان
دانشکده فنی ومهندسی – بخش مهندسی کامپیوتر

درس: آزمایشگاه ریز پردازنده

موضوع :
فاز نهایی پروژه: سیستم هشدار دهنده موانع عقب

استاد: خانم دکتر قزوینی

داشجویان:

علی سلطانی ۹۶۴۰۵۰۴۵
ساناز رستگاری ۹۶۴۰۵۰۴۱

ترم پاییزی ۰۰-۹۹

نام و موضوع پروژه:

سیستم هشدار دهنده موانع عقب

قطعات و موارد استفاده شده

1. برد stm32f103c8tx(blue pill)
2. پروگرامر STLINK V2
3. 2*سنسور التراسونیک HC-SR04
4. برد بورد سوراخدار
5. 2*16 LCD کاراکتری
6. ماژول i2c
7. Led به سه رنگ : سبز ، زرد، قرمز
8. سیم معمولی
9. سیم جامپر به تعداد نیاز
10. STM32CUBEIDE
11. C++
12. کتابخانه i2c LCD

دید کلی مراحل انجام پروژه:

1. نصب برد پردازنده روی برد بورد
2. نصب پروگرامر روی پردازنده
3. نصب ماژول های اولتراسونیک
4. نصب LCD کاراکتری
5. نصب پتانسیومتر و LED های هشدار
6. نصب اتصالات مورد نیاز به کمک سیم ها و سیم های جامپر
7. نوشتن برنامه های اتصالات اولتراسونیک ها و ال سی دی و پردازنده
8. ریختن برنامه روی برد و تست و هماهنگ سازی ماژول

مقدمه:

پروژه ای که در ادامه به روند ساخت و کارکرد آن میپردازیم یک سیستم هشدار دهنده موانع عقب است که از ماژول های اولتراسونیک برای سنجش میزان فاصله تا اجسام استفاده شده است که از ماژول HCSR04 استفاده کردیم که توضیحات بیشتر آن را در فصل دوم شرح میدهیم.

به عنوان ریزپردازنده ازSTM32f103C8t6 استفاده شده که از قابلیت های مهم آن برای این پروژه می توان از قالبیت timer آن نام برد که در ادامه بیشتر به آن میپردازیم.

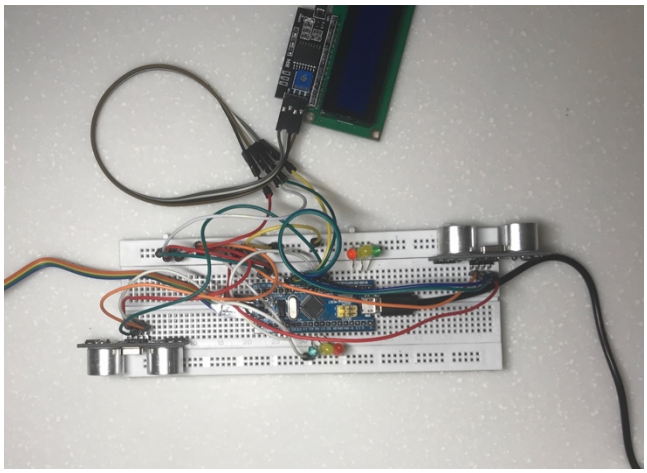
پیش از هرچیز دیگری قصد داریم به شما بگوییم که درنهایت با چه سیستمی رو به رو خواهید شد:

سیستمی که ساخته شده یک سیستم هشدار موانع عقب است که بر اساس اینکه جسم چقدر با شما فاصله دارد اخطار ها و هشدار های متفاوتی را هم به صورت متنی و هم به صورت نوری به شما میدهد:

به طور مثال اگر فاصله جسم از جایی که ماژول فاصله سنج قرار میگیرد بیشتر از ۲۰ سانتی متر باشد بر روی LCD موجود بر روی برد کلمه"safe" چاپ خواهد شد علاوه براین شما با دیدن LED سبز رنگ روشن میتوانید از میزان امن بودن فاصله اتان تا جسم مطمئن شوید.

حال فرض کنید که فاصله شما از جسم کمتر از ۲۰ سانتی متر و بیشتر از ۱۰ سانتی متر است به این هنگام با دیدن LED زرد رنگ و یا دیدن کلمه "WARNING" بر روی LCD به این بازه پی میبرید

اما هنگامی که LED قرمز رنگ و یا کلمه "STOP" را بر روی LCD مشاهده کردید متوجه میشوید که فاصله شما تا جسم کمتر از ۱۰ سانتی متر است.



فصل اول : مسئله timer

مسئله timer همیشه یکی از چالش بر انگیز ترین مسائل در هر میکروکنترلری بوده در این فصل ما یکی از قابلیت های STM32 Timers به نام input capture که استفاده کرده ایم را معرفی میکنیم

همانطور که از نام آن پیداست ، از input capture برای گرفتن سیگنال ورودی داده شده به میکروکنترلر استفاده می شود و فرکانس و عرض پالس آن را اندازه گیری می کند.

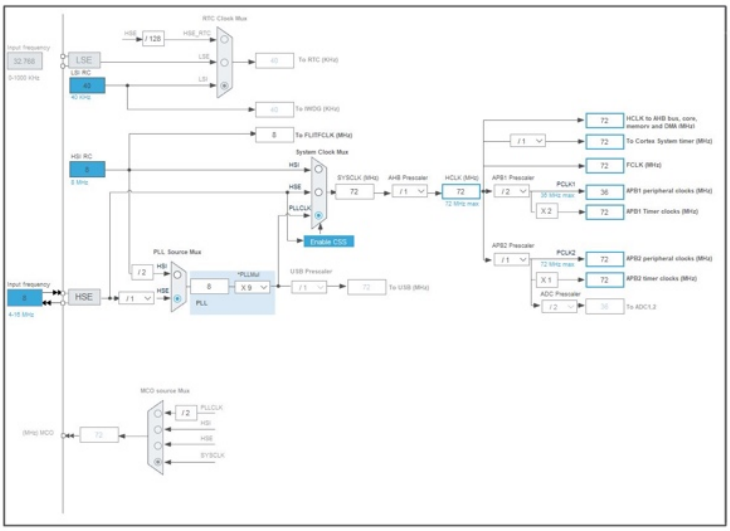
کار کرد این input capture به شرح زیر است:

۱. وقتی که یک لبه بالا رونده کلاک تشخیص داده می شود تابع callback صدا زده میشود

۲. در اینجا ما قصد داریم TIMESTAMP را ضبط کرده و در برخی متغیرها ذخیره کنیم

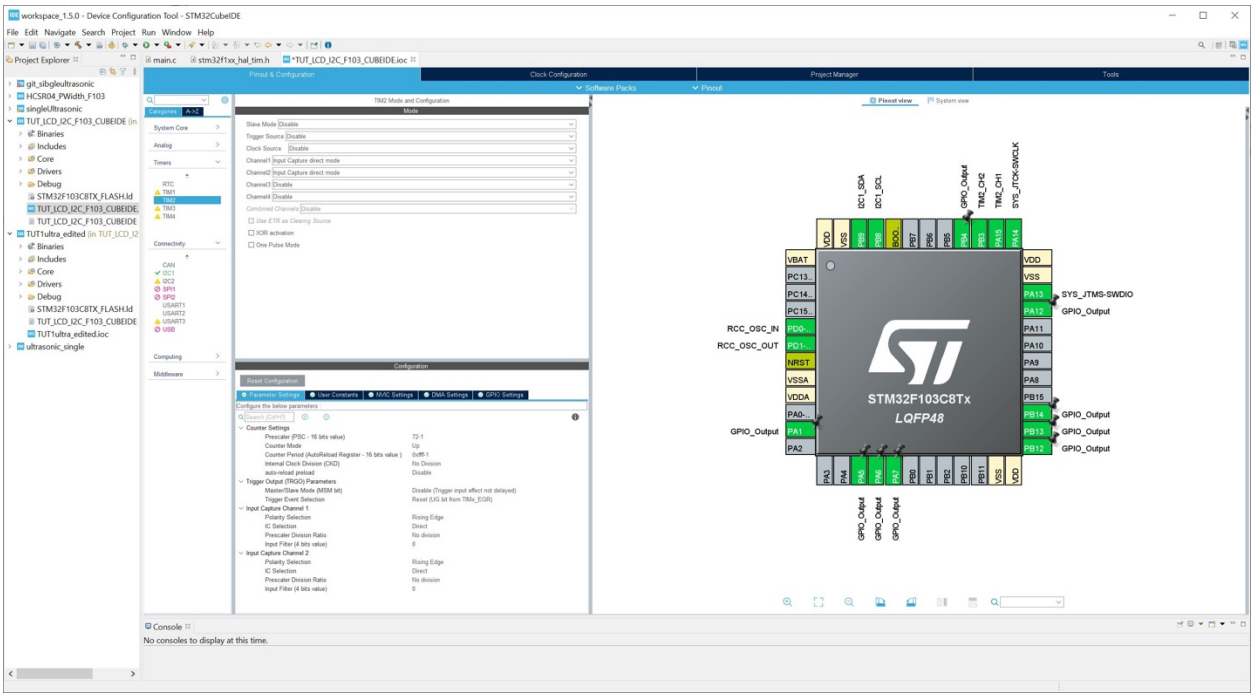
۳. به هنگام تشخیص دومین لبه بالارونده کلاک TIMESTAMP دیگری capture و ذخیره می شود

۴. اکنون می توان فرکانس را با یافتن تفاوت بین این 2 مقدار گرفته شده و تقسیم ساعت تایمر بر این اختلاف محاسبه کرد.



همانطور که در تنظیمات کلاک مشاهده می کنید تمام کلاک های تایمر در ۷۲ مگاهرتز هستند. ما TIM2 را انتخاب کردیم و در ۷۲ مگاهرتز کار می کند.

این قسمت بسیار مهم است. حداقل فرکانسی که دستگاه می تواند بخواند به آن بستگی دارد. در زیر پیکربندی TIM2 آورده شده است که با توجه به اینکه دو ماژول اولتراسونیک داریم از دوکانال ۱ و ۲ استفاده کردیم که به ترتیب مربوط اند به پین های pA15 و pB3.



در بالا می بینید که تنها چیزی که از تنظیمات پیش فرض تغییر داده شده این است که channel 1 و channel 2 ، input capture mode را فعال شده و ARR روی 0Xffff-1 تنظیم شده است. این حداکثر مقداری است که زمان محاسبه می شود.

حداقل فرکانسی که تایمر می تواند بخواند برابر است با (TIMx CLOCK / ARR). در کار ما (۷۲ مگاهرتز / 65536) = ۱۰۹۸ هرتز خواهد بود.
ما می توانیم ساعت TIMx را کاهش دهیم تا حداقل فرکانس را کاهش دهیم اما این همچنین حداکثر فرکانس قابل اندازه گیری را کاهش می دهد. بنابراین باید در مورد دامنه اینجا بسیار منطقی رفتار کرد.

نگاهی بر کد:

```
// Let's write the callback function

void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim)
{
    if (htim->Channel == HAL_TIM_ACTIVE_CHANNEL_1)  // if the interrupt source is channel1
    {
        if (Is_First_Captured==0) // if the first value is not captured
        {
            IC_Val1 = HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_1); // read the first value
            Is_First_Captured = 1;  // set the first captured as true
            // Now change the polarity to falling edge
            __HAL_TIM_SET_CAPTUREPOLARITY(htim, TIM_CHANNEL_1, TIM_INPUTCHANNELPOLARITY_FALLING);
        }
        else if (Is_First_Captured==1)   // if the first is already captured
        {
            IC_Val2 = HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_1);  // read second value
            __HAL_TIM_SET_COUNTER(htim, 0);  // reset the counter

            if (IC_Val2 > IC_Val1)
            {
                Difference = IC_Val2-IC_Val1;
            }

            else if (IC_Val1 > IC_Val2)
            {
                Difference = (0xffff - IC_Val1) + IC_Val2;
            }

            Distance = Difference * .034/2;
            Is_First_Captured = 0; // set it back to false

            // set polarity to rising edge
            __HAL_TIM_SET_CAPTUREPOLARITY(htim, TIM_CHANNEL_1, TIM_INPUTCHANNELPOLARITY_RISING);
            __HAL_TIM_DISABLE_IT(&htim2, TIM_IT_CC1);
        }
    }
```

Problems Tasks Console Properties

Property	Value
Info	
derived	false

C_F103_CUBEIDE/Core/Src/main.c - STM32CubeIDE

gate Search Project Run Window Help

main.c stm32f1xx_hal_tim.h TUT_LCD_I2C_F103_CUBEIDE.ioc

```
        }
    }

    if (htim->Channel == HAL_TIM_ACTIVE_CHANNEL_2)  // if the interrupt source is channel2
        {
            if (Is_First_Captured_1==0) // if the first value is not captured
            {
                IC_Val1_1 = HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_2);// read the first value
                Is_First_Captured_1 = 1;// set the first captured as true
                // Now change the polarity to falling edge
                __HAL_TIM_SET_CAPTUREPOLARITY(htim, TIM_CHANNEL_2, TIM_INPUTCHANNELPOLARITY_FALLING);
            }
            else if (Is_First_Captured_1==1)   // if the first is already captured
            {
                IC_Val2_1 = HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_2); // read second value
                __HAL_TIM_SET_COUNTER(htim, 0);  // reset the counter

                if (IC_Val2_1 > IC_Val1_1)
                {
                    Difference_1 = IC_Val2_1-IC_Val1_1;
                }

                else if (IC_Val1_1 > IC_Val2_1)
                {
                    Difference_1 = (0xffff - IC_Val1_1) + IC_Val2_1;
                }

                Distance_1 = Difference_1 * .034/2;
                Is_First_Captured_1 = 0; // set it back to false

                // set polarity to rising edge
                __HAL_TIM_SET_CAPTUREPOLARITY(htim, TIM_CHANNEL_2, TIM_INPUTCHANNELPOLARITY_RISING);
                __HAL_TIM_DISABLE_IT(&htim2, TIM_IT_CC2);
            }

        }
}
```

تابع callback بالا هرگاه لبه بالا رونده کلاک پیش آید صدا زده میشود اگر Is_First_Captured برابر صفر باشد از این رو مقدار IC_Val1 ضبط میشود هنگامی که برای بار دوم به لبه بالا رونده کلاک برسیم یعنی Is_First_Captured برابر یک باشد آنگاه IC_Val2 ضبط می شود و مقدار تفاوت این دو در Difference و در نهایت با توجه به نحوه ی محاسبه فرکانس که دربالاتر ذکر شد مقدار frequency محاسبه میشود.

در تابع main برای شروع TIMER در input capture interrupt mode از کد زیر استفاده میکنیم:

```
HAL_TIM_IC_Start_IT(&htim۲, TIM_CHANNEL_1);
HAL_TIM_IC_Start_IT(&htim۲, TIM_CHANNEL_2);
```

هنگام لبه بالارونده اولین Timestamp را ضبط میکند و حال پلاریته روی لبه پایین رونده قرار میگیرد
دومین Timestamp زمانی ضبط میشود که به لبه پایین رونده کلاک رسیده باشیم
تفاوت میان این دو Timestamp محاسبه میشود(difference) که مقدارش به میکروثانیه است در حالی که تایمر با 1 MHz مشغول فعالیت است
بر پایه مقدار difference مقدار distance بر اساس فرمولی که در datasheet داده شده محاسبه میشود:
در نهایت هم interrupt ، disable می شود تا سیگنال های ناخواسته ای ضبط نشود

لازم به ذکر است که به ازای هر سنسور اولتراسونیک ،باید کد تغییر داده شده مختص آن سنسور مربوط به کانال سنسور که در ابتدا در شرط if ذکر شده تکرار شود.

فصل دوم: ماژول الترا سونیک:

ماژول اولتراسونیک HC - SR04 عملکرد اندازه گیری غیر تماسی 2 سانتی متر - 400 سانتی متر را فراهم می کند.
ماژول ها شامل فرستنده های اولتراسونیک ، گیرنده و مدار کنترل هستند.
در این فصل نحوه اتصال ماژول حسگر اولتراسونیک HCSR04 را با STM32 بررسی می کنیم.
ماژول اولترا سونیک به دو پایه ی trigger و echo نیاز دارد که پایه های echo را برای دو ماژول در فصل اول مشخص کردیم
حال دو پایه ی pB4 و pA12 را برای trigger فعال میکنیم.

نحوه کارکرد ماژول:

ماژول یک ultrasound فرکانسی را با فرکانس 40 کیلوهرتز ساطع می کند که پس از بازتاب از مانع ، به ماژول باز می گردد.
با استفاده از زمان حرکت و سرعت صدا می توان فاصله سنسور و مانع را محاسبه کرد.

عملکرد ماژول

با توجه به hc-sr04 datasheet ، موارد زیر لازم است انجام شود:

- برای حداقل ۱۰ میکروثانیه باید پین trig، high شود.
- ماژول اکنون 8 سیکل از ultrasound را با فرکانس 40 کیلوهرتز ارسال می کند و تشخیص می دهد که آیا سیگنال پالس برگشت دارد یا خیر

اگر سیگنال بازگشت داشته ،ماژول یک high pulse را تولید می کند که عرض آن متناسب با دامنه جسم خواهد بود

- میزان فاصله با توجه به فرمول زیر محاسبه می شود:

  Range= high level time* velocity(340m/s)/2

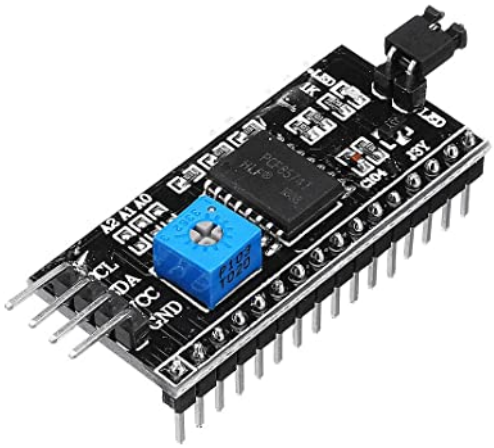
- پیشنهاد شده که به ازای هر بار تکرار حلقه حداقل۶۰ میلی ثانیه تا تکرار بعدی(شروع دوباره کار) به ماژول فرصت داده شود که با توجه به داشتن LCD و LEDها احتیاج به زمان بیشتری در کار داشتیم و زمان آن را با آزمایشات چندباره ۵۰۰ میلی ثانیه قرار دادیم.

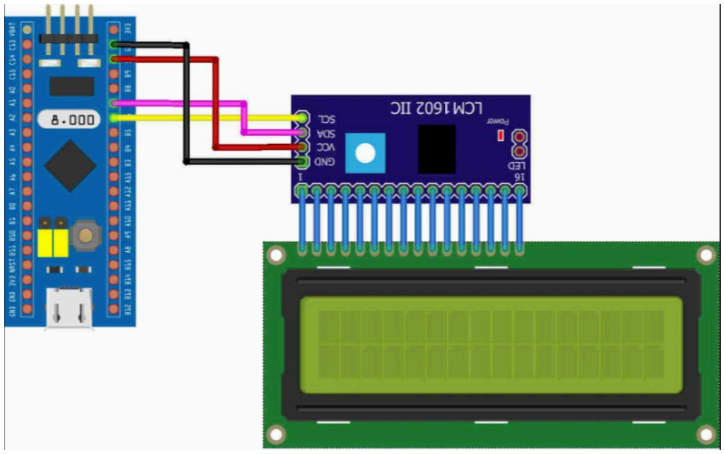
```
158
159 void HCSR04_Read (void)
160 {
161     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_12, GPIO_PIN_SET);  // pull the TRIG pin HIGH
162     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, GPIO_PIN_SET);  // pull the TRIG pin HIGH
163     delay(10); // wait for 10 us
164     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_12, GPIO_PIN_RESET);  // pull the TRIG pin low
165     HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, GPIO_PIN_RESET);  // pull the TRIG pin low
166
167     __HAL_TIM_ENABLE_IT(&htim2, TIM_IT_CC1);
168     __HAL_TIM_ENABLE_IT(&htim2, TIM_IT_CC2);
169 }
170
```

HCSR04_Read-> سنسور را برای شروع اندازه گیریtrigged میکند.

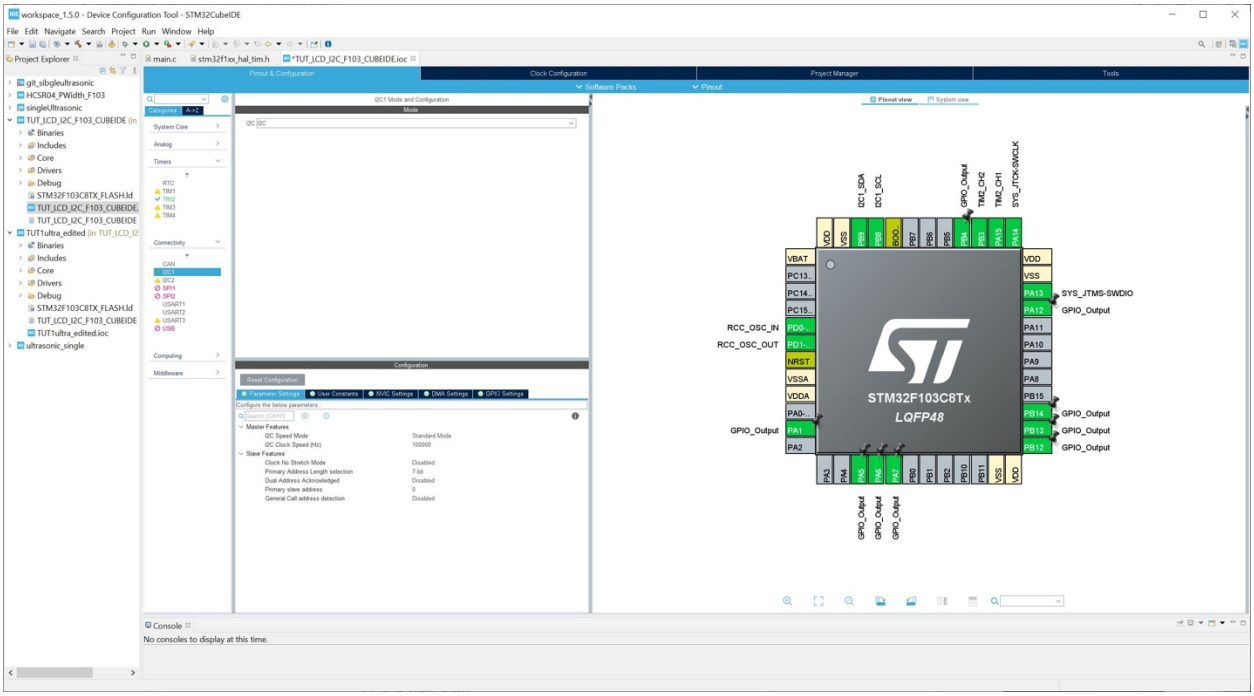
- پین های trig را برای ۱۰ میکروثانیه high میکند و سپس آن ها را low میکند.
- این حرکت ماژول ها را وادار به شروع اندازه گیری میکند و سنسورهای -echo pin را برای مدت زمانی high میکند.
- برای اینکه time را اندازه بگیریم timer interrupt را enable میکنیم تا بتوانیم لبه های بالارونده و پایین رونده را ضبط کنیم.

فصل سوم : اتصال ICD



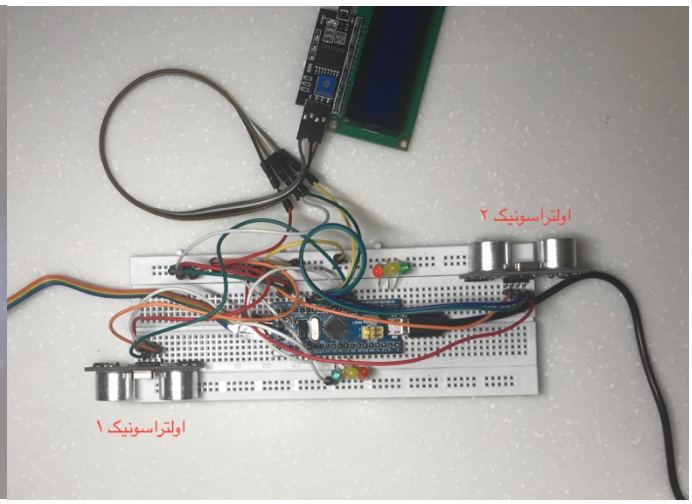


همانطور که در بالا مشاهده می کنید PCF8574 دارای 4 پایه ورودی GND، VCC، SDA، SCLو ۱۶ پایه برای اتصال به LCD کارکتری است. ما LCD خود را به این 16 پایه متصل خواهیم کرد.



با فعال کردن گزینه i2c در قسمت connectivity>i2c1 دو پایه از میکرو به طور اتوماتیک فعال می شوند(pB8 به عنوان i2c-1 SCL و pB9 به عنوان i2c-1SDA)

خط اول LCD مربوط به نمایش اطلاعات ماژول التراسونیک ۱ و خط دوم مربوط به اولترا سونیک ۲ می باشد.



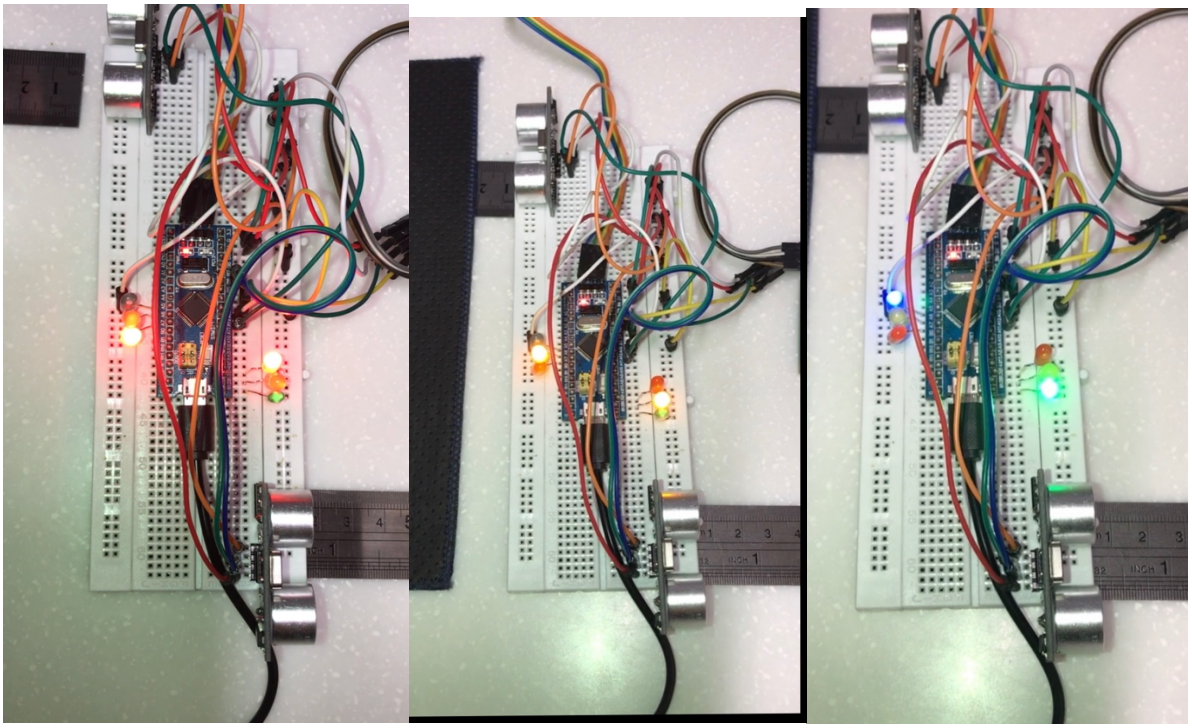
فصل چهارم: سیستم هشدار موانع:

همانطور که قبل تر ذکر شد برای هشدار از دو روش متنی و نوری برای سیستم هشدار دهنده استفاده کرده ایم:

رنگ سبز: فاصله بیشتر از ۲۰ سانتی متر + کلمه "safe!"

رنگ زرد: حد فاصل بین ۱۰ الی ۲۰ سانتی متر + کلمه"WARNING!!!"

رنگ سبز: فاصله کمتر از ۱۰ سانتی متر + کلمه "STOP!"



نگاهی برکد:

به ازای هر بازه ای که نیاز است اطلاعات (متنی و نوری) عوض شوند در هر شرط برای اطمینان از خاموش شدن سایر LED ها ابتدا همه LED هارا reset میکنیم سپس LCD cursor را بر روی جای مناسبش (بسته به اینکه اطلاعات کدام ماژول را میخواهد بنویسد) قرار میدهیم ،هشدار متنی مربوطه به LCD فرستاده میشود و سپس LED مورد نظر set میشود.

```
210   lcd_init ();
211
212   lcd_send_string ("HELLO");
213
214   HAL_Delay(1000);
215
216   lcd_put_cur(1, 0);
217
218   lcd_send_string("from the outside");
219
220   HAL_Delay(2000);
221
222   lcd_clear ();
223
```

```
227       HCSR04_Read();
228       lcd_put_cur(0, 0);
229       lcd_send_data((Distance/100) + 48);   // 100th pos
230       lcd_send_data(((Distance/10)%10) +48);  // 10th pos
231       lcd_send_data((Distance%10)+48);  // 1st pos
232       lcd_send_string(" cm-1");
233
234       if ( Distance >= 20 )
235       {
236           HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_RESET);
237           HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_RESET);
238           HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_RESET);
239           lcd_put_cur(0, 8);
240           lcd_send_string(">SAFE");
241           //HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, GPIO_PIN_SET);
242           HAL_GPIO_TogglePin(GPIOA,GPIO_PIN_5);
243       }
244       else
245       {
246           if (Distance >= 10 )
247           {
248               HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_RESET);
249               HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_RESET);
250               HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_RESET);
251               lcd_put_cur(0, 8);
252               lcd_send_string(">WARNING");
253               HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_SET);
254           }
255           else
256           {
257               HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, GPIO_PIN_RESET);
258               HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, GPIO_PIN_RESET);
259               HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_RESET);
260               lcd_put_cur(0, 8);
261               lcd_send_string(">STOP");
262               HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, GPIO_PIN_SET);
263           }
264       }
```

```
266       lcd_put_cur(1, 0);
267       lcd_send_data((Distance_1/100) + 48);   // 100th pos
268       lcd_send_data(((Distance_1/10)%10) +48);  // 10th pos
269       lcd_send_data((Distance_1%10)+48);  // 1st pos
270       lcd_send_string(" cm-2");
271
272       if ( Distance_1 >= 20 )
273       {
274           HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, GPIO_PIN_RESET);
275           HAL_GPIO_WritePin(GPIOB, GPIO_PIN_13, GPIO_PIN_RESET);
276           HAL_GPIO_WritePin(GPIOB, GPIO_PIN_14, GPIO_PIN_RESET);
277           lcd_put_cur(1, 8);
278           lcd_send_string(">SAFE");
279           //HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, GPIO_PIN_SET);
280           HAL_GPIO_TogglePin(GPIOB,GPIO_PIN_12);
281       }
282       else
283       {
284           if ( Distance_1 >= 10 )
285           {
286               HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, GPIO_PIN_RESET);
287               HAL_GPIO_WritePin(GPIOB, GPIO_PIN_13, GPIO_PIN_RESET);
288               HAL_GPIO_WritePin(GPIOB, GPIO_PIN_14, GPIO_PIN_RESET);
289               lcd_put_cur(1, 8);
290               lcd_send_string(">WARNING");
291               HAL_GPIO_WritePin(GPIOB, GPIO_PIN_13, GPIO_PIN_SET);
292           }
293           else
294           {
295               HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, GPIO_PIN_RESET);
296               HAL_GPIO_WritePin(GPIOB, GPIO_PIN_13, GPIO_PIN_RESET);
297               HAL_GPIO_WritePin(GPIOB, GPIO_PIN_14, GPIO_PIN_RESET);
298               lcd_put_cur(1, 8);
299               lcd_send_string(">STOP");
300               HAL_GPIO_WritePin(GPIOB, GPIO_PIN_14, GPIO_PIN_SET);
301           }
```