دانشکده مهندسی کامپیوتر و فناوری اطلاعات

GROUP MEMBERS:

KASRA KHALAFI: 9523038

SOBHAN RADFAR: 9523044

GOLNAZ BASHIRIAN:9523435

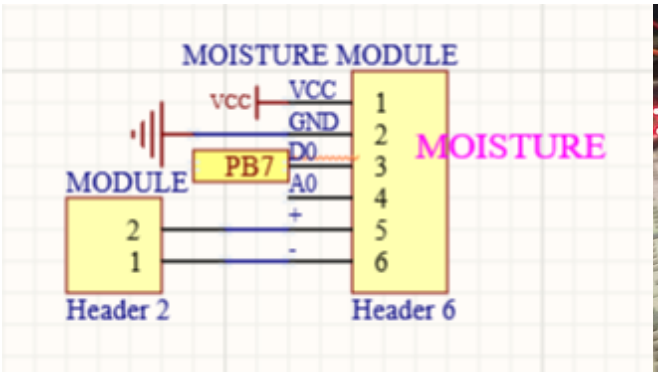
Home Work 2

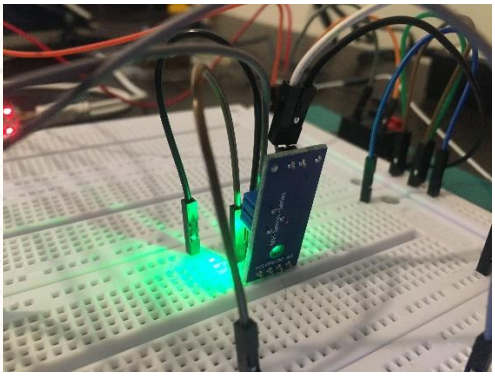
Microprocessors 1

Dr.Saeed Sharifian Khortoomi

بخش ۱: ماژول رطوبت سنج

در این بخش تنها از پین DIGITAL مدار رطوبت سنج برای اتصال به پین PB7 استفاده شد که این پین به صورت INPUT در CubeMX تعریف شده بود. همچنین زمین و VDD هم به زمین و منبع تغذیه وصل شد. در صورتی که رطوبت از حد معینی بالا تر میرفت این پین مقدار صفر را به ما باز میگرداند که نشان از رطوبت خاک داشت و به کمک یک مقاومت متغیر این میزان رطوبت میتوانست مشخص شود. در این بخش به مشکل خاصی برخورده نشد و به راحتی توانسته شد این پین خوانده شود.

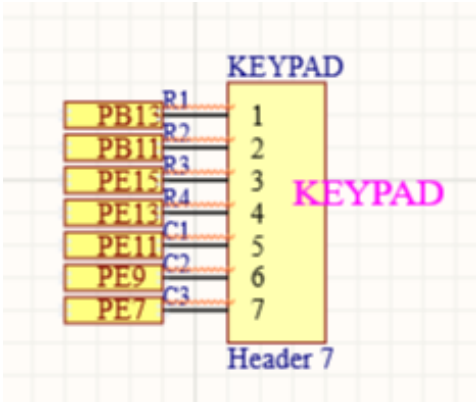


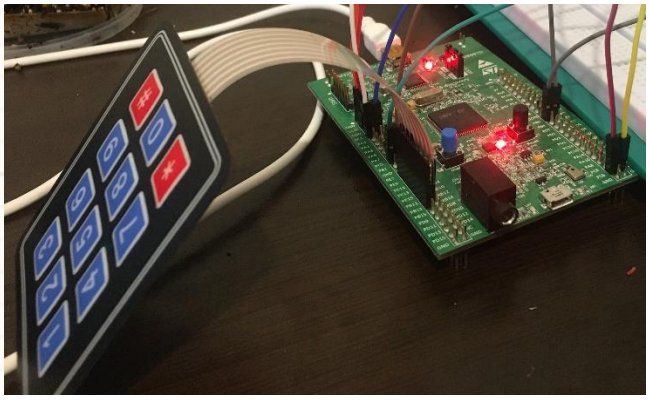


بخش ۲: کیپد (Keypad)

قسمت keypad ما دارای ۴ سطر و ۳ ستون بود که در نتیجه از ۷ سیم برای مشخص شدن کلید فشرده شده استفاده میشد بدین صورت که ۴ عدد سیم به ۴ عدد پین GPIO OUTPUT متصل میشد و سه سیم دیگر به ۳ عدد پین GPIO INPUT متصل میشد که بسته به HIGH و LOW شدن سطر وستون ها، کلید فشرده شده مشخص میشد. مشکل اصلی ما در این بخش اتصال به پین های یاد شده در صورت سوال و قطعی سیم های کیپدمان بود. با توجه به اینکه پین های مدار برای وصل شدن کنار هم نبودند نیاز به سیم برای وصل کردن کیپد به board بود که این سیم های اضافی باعث قطعی و وصلی در کیپد میشدند در نتیجه از پین های دیگری برای کنترل کیپد استفاده شد. پین های سطر ها به ترتیب PB13، PB11، PE15 و PE13 بودند و پین های ستون به ترتیب PE11، PE9 و PE7 بودند.

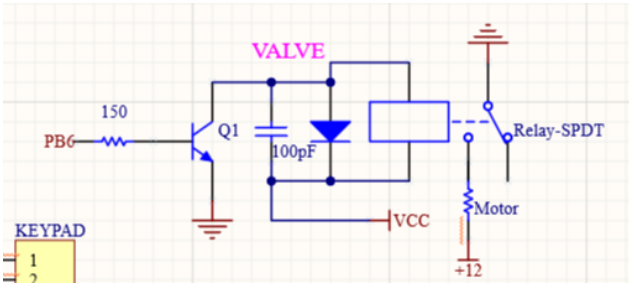
نحوه ی چک کردن کلید فشرده شده هم بدین صورت بود که ابتدا سطر ها را چک میکردیم بدین ترتیب که ابتدا به صورت 0111 پین های سطر را به ترتیب مقدار دهی میکردیم که این عدد نشانگر سطر اول بود سپس مقدار پین های ستون های ۱ تا ۳ را میخواندیم. در صورتی که ستونی صفر شده بود، یعنی آن ستون انتخاب شده بود و با توجه به انکه در سطر یک بودیم با توجه به ستون دکمه ی فشرده شده مشخص میشد. در صورتی که ستونی در سطر یک فشرده شده نبود، عدد 0111 را به صورت دایره ای شیفت راست میدادیم و که در این صورت برابر عدد 1011 و سطر دوم میشد. در سطر دوم نیز چک میشد که کدام ستون فشرده شده است و در صورتی که ستونی فشرده نشده بود به سطر بعدی با عدد دهی 1101 به سطر ها میرفتیم و در صورتی که در این سطر نیز دکمه ای فشرده نشده بود به سطر اخر با مقدار دهی 1110 میرفتیم و به چک کردن فشرده شدن دکمه از طریق مقدار high و low بودن ستون ها میپرداختیم.

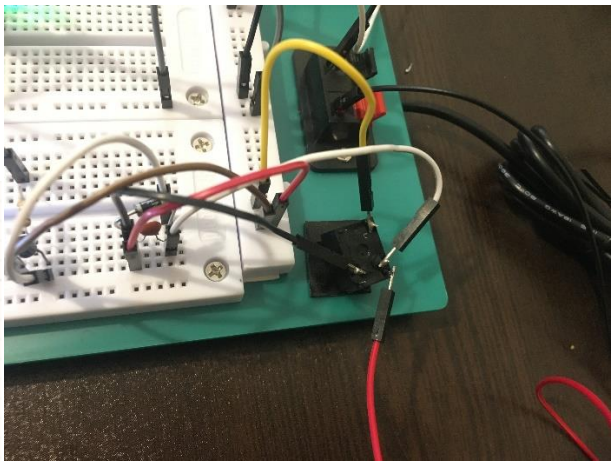




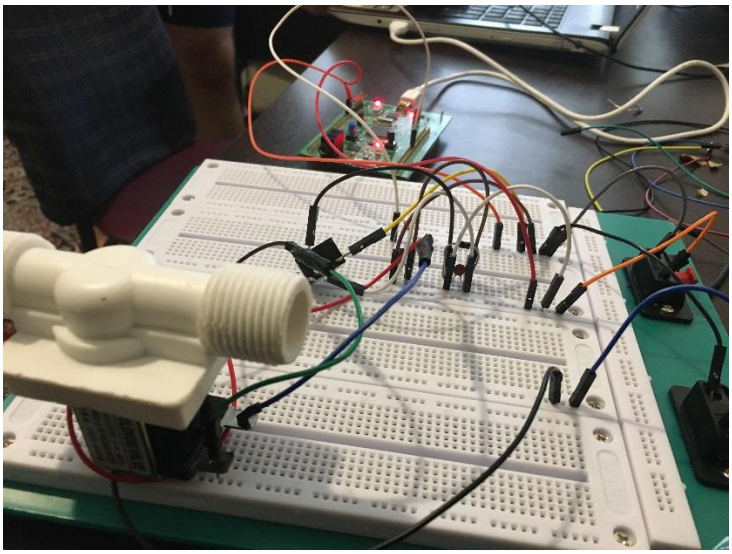
بخش۳: شیر آب الکتریکی (VALVE)

شیر آب الکتریکی استفاده شده دارای ولتاژ مورد نیاز ۱۲ ولت و توان خروجی ۴٫۸ می باشد. از آنجایی که ولتاژ پین های میکرو نمیتواند ولتاژ ۱۲ ولت را تامین کند در نتیجه از یک مدار جدا برای راه اندازی این موتور باید استفاده شود. برای راه اندازی موتور و شیر، از یک رله ی ۱۲ ولتی کمک گرفته شد که با یک خازن و یک دیود موازی شده بود. در صورت تامین ولتاژ لازم برای شروع به کار در دو سر رله، رله بسته میشد(مسیر اتصالش عوض میشد) و مقدار ۱۲ ولت دو سر موتور می افتاد و موتور شروع به کار میکرد و در نتیجه به برد فشاری وارد نمیشد. مشکلی که ما در این بخش داشتیم این بود که رله به دلیل ناتوانی در تامین ولتاژ مورد نیاز برای تحریک، نمیتوانست بسته شود در نتیجه از یک رله ی ۵ ولتی برای تحریک استفاده شد که مشکل ۱۲ ولت را حل کرد. شکل زیر، نحوه ی بسته شدن مدار را نشان میدهد:



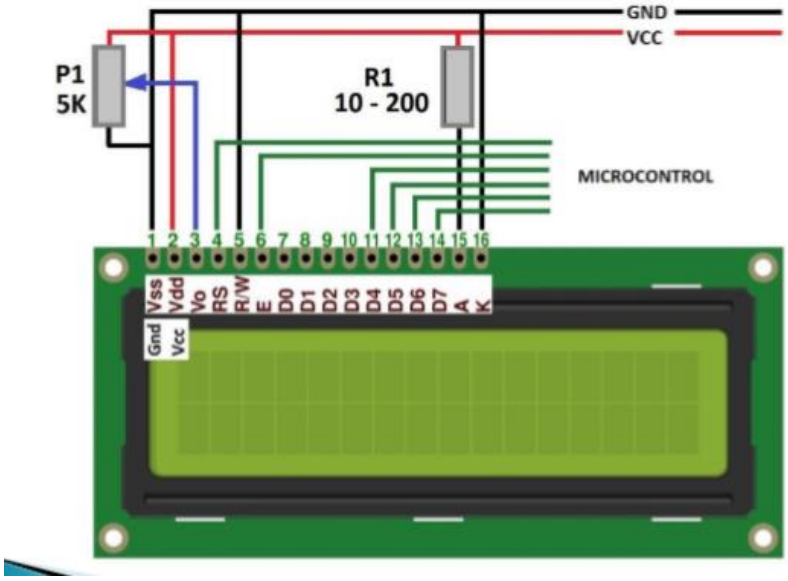


شیر بسته شده به مدار نیز به شکل زیر میباشد:



بخش ۴: نمایشگر ال سی دی(LCD)

ال سی دی مورد استفاده ۲ در ۱۶ می باشد که دارای ۱۴ پایه می باشد. شکل کلی اتصال ال سی دی به صورت زیر می باشد:



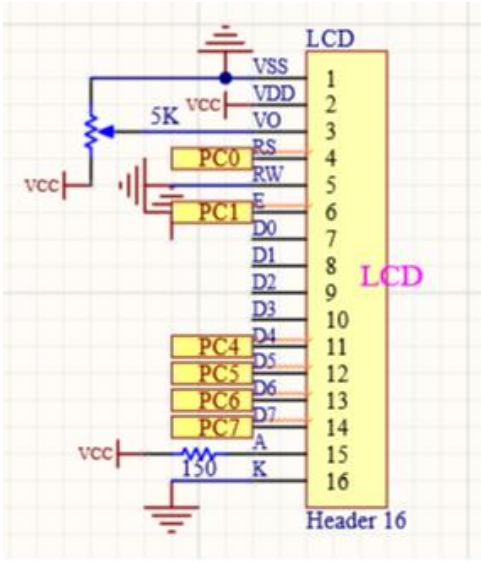
پایه های RS و enable ال سی دی به ترتیب به پین های PC0 و PC1 متصل شدند. پین RS که مخفف Register Select می باشد برای انتخاب مد استفاده می شود که انتخاب کننده ی مد دستورات یا مد کاراکتری است. پین های D4 تا D7 نیز به ترتیب به پین های PC4 تا PC7 میکرو متصل شدند. برای ارسال اطلاعات از ۴ پین به جای ۸ پین استفاده شد که باعث می شود از پین های کمتری برای نمایش داده ها روش ال سی دی استفاده کنیم.

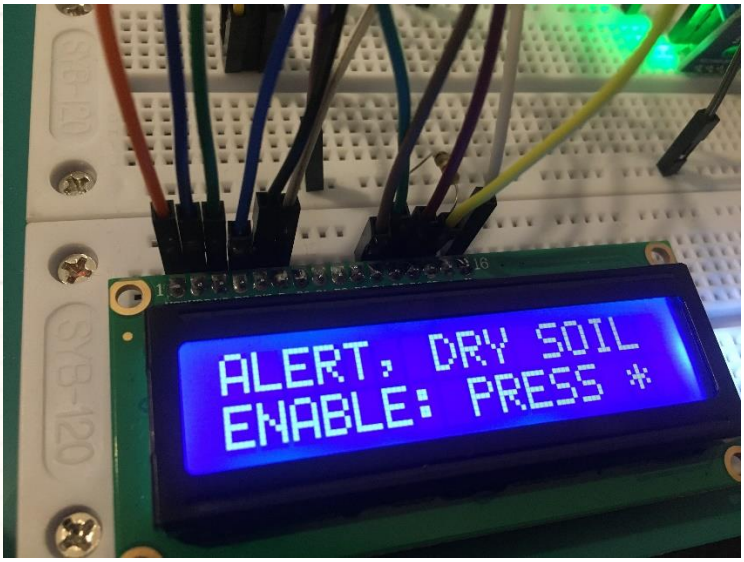
برای قسمت کد، یک کتابخانه با نام lm016.h ساخته شد که دستورات مربوط به نمایش دادن روی ال سی دی در این کتابخانه قرار دارد.

دستورات تعریف شده در این کتابخانه عبارتند از:

```
static void lcd_send_4bit(uint8_t data);          // SENDING 4 BIT OF DATA
static void lcd_send(int8_t rs,uint8_t data);     // SENDIGN ALL DATA USING lcd_send_4bit() FUNCTION
static void delay_us(uint32_t delay);             // DELAY TO us
void lcd_data(char c);                            // GETS THE CHARACTERS AND PASS IT TO THE lcd_send() FUNCTION
void lcd_init(void);                              // INITIALIZING THE LCD
void lcd_clr(void);                               // CLEARING LCD
void lcd_gotoxy(char x, char y);                  // SETTING CURSER POSITION
void lcd_puts(char *text);                        // PRINTING GIVEN TEXT IN THE FUNCTION ON LCD
```

توضیحات و روش انجام هر یک از دستورات نیز به صورت کامنت در کد ها موجود می باشد.





بخش ۵:

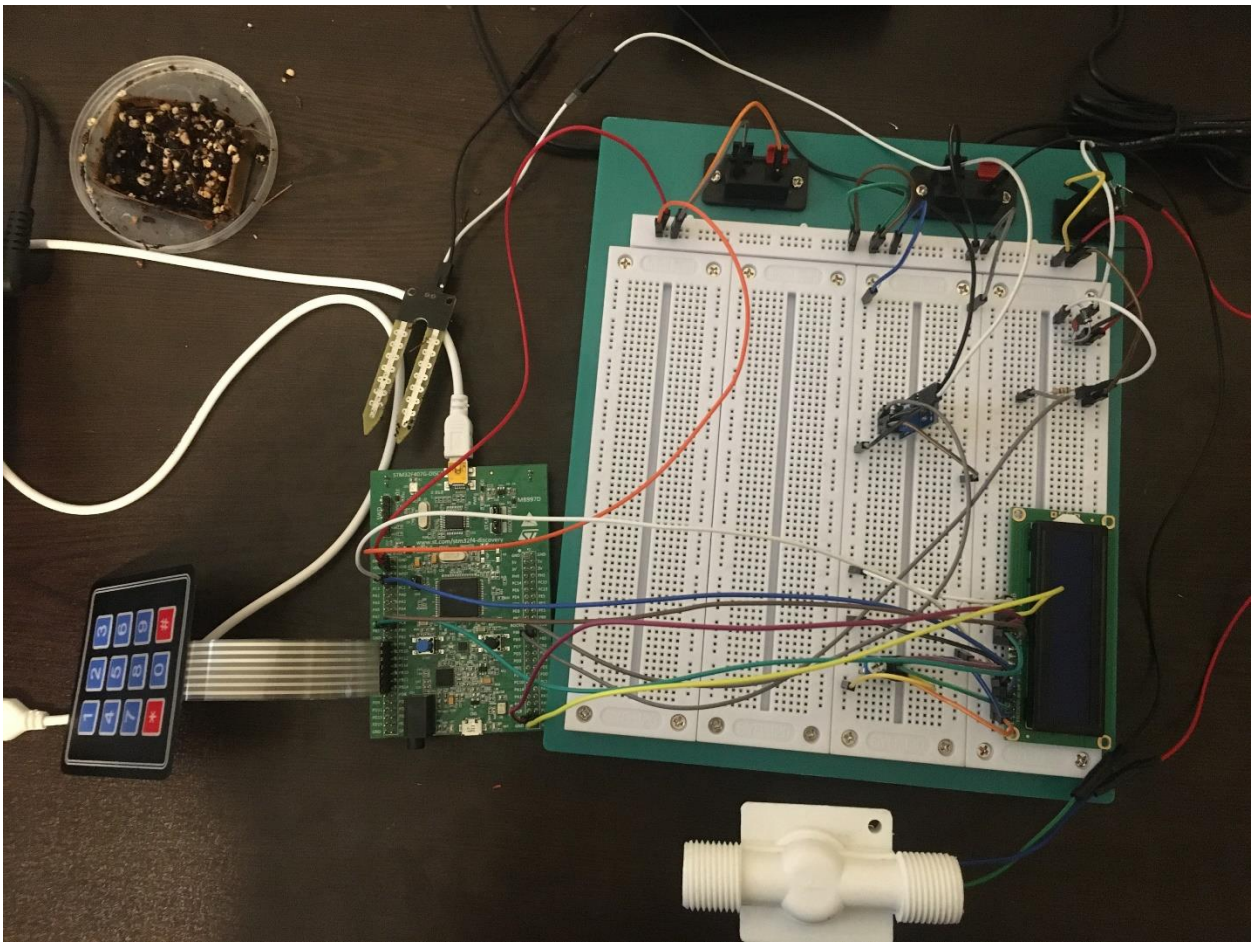
این بخش تجمیع کل بخش های بالایی پیاده سازی شده است تا بتوان به آبیاری هوشمند خاک پرداخت.

آبیاری بدین صورت است که ابتدا چک میکند که آیا خاک خشک است یا تر. در صورت خشک بودن خاک، یک آلارم در LED روی برد روشن میشود و روی LCD متن Alarm چاپ میشود و نوشته میشود که برای مشخص کردن مدت زمان آبیاری کلید ستاره (*) فشرده شود. در صورت فشردن کلید ستاره، شکل زیر روی ال سی دی ظاهر میشود:

زمان باز بودن شیر انتخاب میشود که این مقدار داخل متغیر t_open ذخیره میشود. پس از انتخاب از بین گزینه ها، شیر به اندازه ی مورد نظر باز می ماند و LED که برای ALARM روشن شده بود خاموش میشود. پس از تمام شدن زمان مورد نظر شیر آبیاری بسته می شود. پس از بسته شدن سپس چک میشود که آیا باز خاک خشک است یا خیر در صورت خشک نبودن شیر باز نمیشود و هر پنج ثانیه یک بار تر بودن یا خشک بودن چک میشود.

برای پیاده سازی مقدار اولیه و default باز بودن نیز ۱ دقیقه در نظر گرفته شده است. میتوانستیم چک کردن و مقدار دهی را به صورت متفاوت نیز انجام دهیم بدین صورت که در هنگام بسته بودن شیر و تر بودن خاک مقدار باز بودن در صورت خشکی بیان کنیم.

با توجه به اینکه به صورت خیلی دقیق گفته نشده بود که چگونه پیاده سازی شود به همین دلیل این روش پیاده سازی انتخاب شد.



پاسخ به سوالات پرسیده شده در سوالات:

- حل مشکل قسمت ۲:

برای حل این مشکل از یک while استفاده میکنیم بدین صورت که شخص تا انگشتش را از روی دکمه برندارد دستوری انجام نشود.بدین ترتیب که ابتدا به کمک سطر و ستون پین تشخیص داده می شود و تا زمانی که دست کاربر از دکمه ی مورد نظر برداشته نشود دستوری برای اجرا ارسال نمی شود.

- با توجه به کدی که نوشتید به این سوال پاسخ دهید که آیا هنگامی که شیر آب باز است، میکروکنترلر میتواند فشرده شدن کلیدی را در کیپد متوجه شود؟ توضیح دهید.

در کدی که نوشته ایم، در حین باز بودن شیر نمیتوان چک کرد که ایا کلیدی فشرده شده است یا خیر زیرا از HAL_DELAY() برای این زمان باز بودن استفاده کرده ایم. روش دیگری که میتوانستیم استفاده کنیم این بود که از یک حلقه ی While استفاده کنیم و به کمک HAL_GetTick() زمان باز بودن شیر را چک کنیم و داخل این حلقه فشرده شدن کلید را چک کنیم. مشکل این روش این است که در صورتی که دکمه ای فشرده شود و برای مثال ۱۰ ثانیه این کار طول بکشد، دیگر زمان باز بودن شیر ۱ دقیقه یا زمان مشخص شده نیست و ۱۰ ثانیه ی فشرده شده کلید نیز به آن اضافه خواهد شد.

- به نظرتان ایراد عمده ی استفاده از توابع delay به غیر از دقیق نبودن چه چیزی میتواند باشد؟

مشکل اصلی استفاده از delay این است که میکرو به صورت کلی درگیر این فانکشن میشود و نمیتواند کار های دیگری انجام بدهد و cpu درگیر انتظار میشود. برای حل این مشکل می توان از INTERRUPT استفاده کرد به طوری که تا زمانی که INTERRUPT نیامده باشد پردازنده به رسیدن به کار های لازم بپردازد و در صورت که INTERRUPT آمد پردازنده شروع به سرویس دهی به روتین وقفه باشد.

- مقادیر بخش سه را بنویسید:

مقاومت بیس استفاده شده برابر با ۱۵۰ اهم می باشد. جریان بیس به صورت زیر محاسبه میشود:

$3.3 - Ib * 0.15 - 0.6 = 0 \rightarrow Ib = 18$ میلی آمپر

میتوان با افزایش مقاومت جریان گذرنده از بیس را کمتر کرد. برای محافظت از مدار و سیم پیچ بوبین در هنگام قطع کردن، از یک دیود و خازن با مقدار ۱۰۰ نانو فاراد به صورت موازی استفاده کرد. توضیحات تکمیلی هم در باره ی نحوه ی بسته شدن نیز در قسمت ۳ توضیح داده شده است.