

به نام خداوند جان و خرد

درس ابزار دقیق

گروه کنترل



مدرس: محمدرضا نیری

پروژه

نیمسال دوم ۱۴۰۱-۱۴۰۲

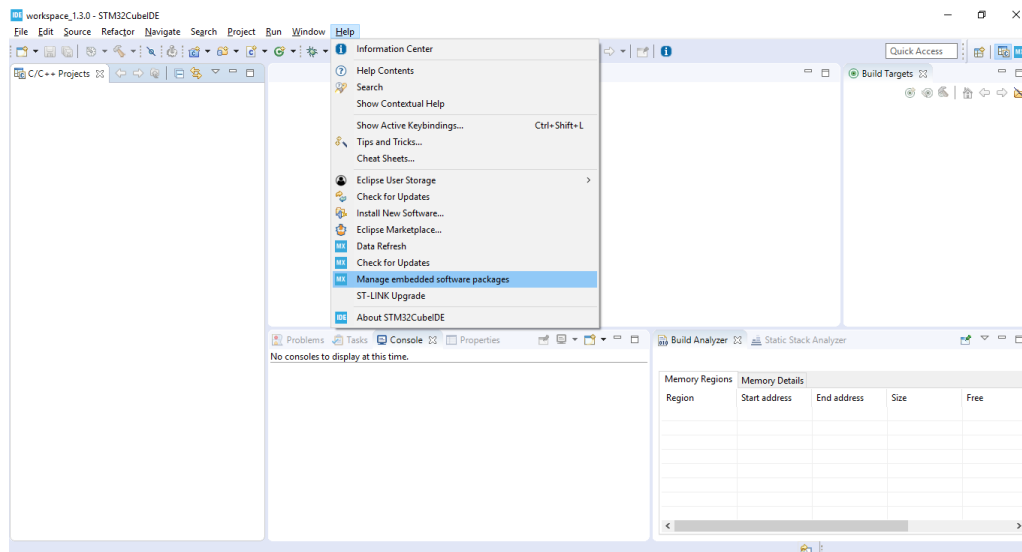
در این مینی پروژه مراحل لازم جهت شروع کار و برنامه نویسی با میکرو کنترلر ARM را دنبال خواهید کرد. برای این منظور از نرم افزار **STM32CubeIDE** و پروتئوس استفاده خواهید کرد. این پروژه نیاز به گزارش نخواهد داشت و تنها ارسال فایل های خواسته شده در هر بخش کفایت می کند. در صورت لزوم پروژه انجام شده با استفاده از اسکایپ ارائه خواهد شد. با مشاهده ویدئوهای مربوط به درس و توضیحات ارائه شده در پروژه نیازی به منابع دیگر نخواهید داشت اما در صورت لزوم می توانید به ویدئوهای کانال زیر نیز مراجعه فرمایید:

<https://www.youtube.com/channel/UC-CuJ6qKst9-8Z-EXjoYK3Q/videos>

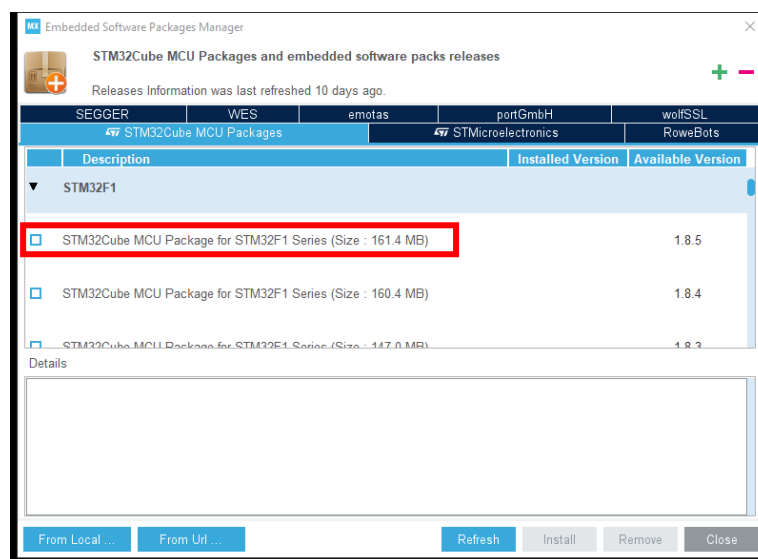
➤ نصب و آماده سازی نرم افزارها

مراحل زیر را پی بگیرید

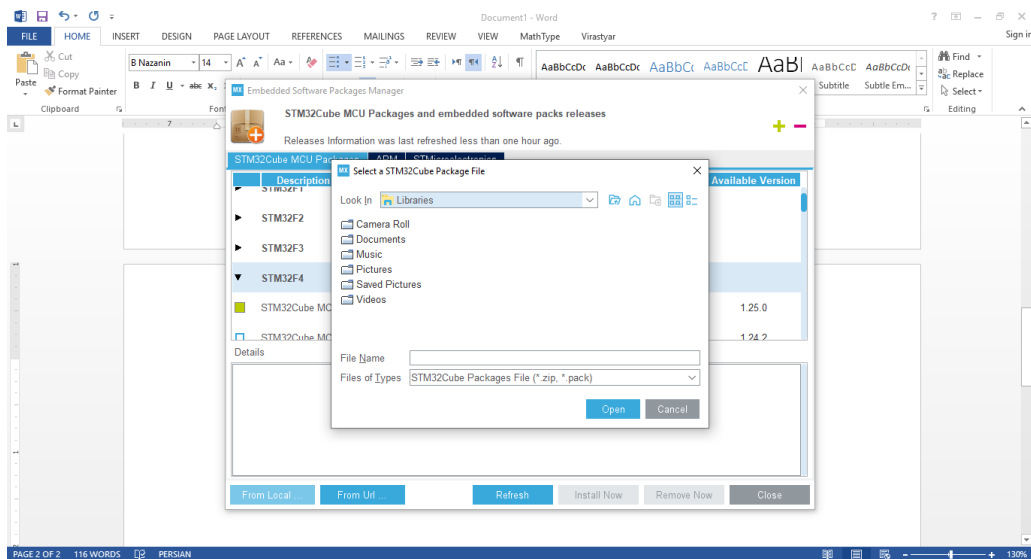
- ۱- نرم افزار STM32CubeIDE را نصب کنید (ویدئو مربوط به چگونگی نصب در سامانه درس بارگذاری شده است)
- ۲- پس از نصب نرم افزار را باز کنید و مطابق شکل زیر پکیج STM32F1 را به یکی از دو روش زیر نصب کنید



➤ دانلود و نصب به صورت آنلاین: که باید به صورت شکل زیر ابتدا تیک مربوط به پکیج مورد نظر را فعال کرده و سپس بر روی گزینه Install Now کلیک کنید

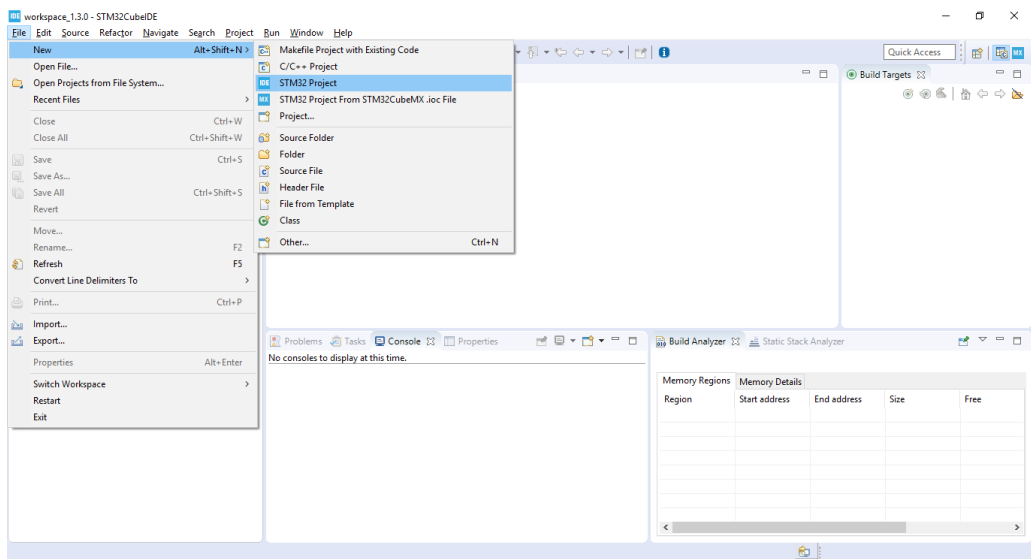


➤ نصب به صورت آفلاین: در صورتی که پکیج مورد نظر رو قبلا دانلود کرده اید کافیست از From Local فایل مورد نظر را بر روی هارد خود پیدا کرده و انتخاب کنید.



پس از نصب درست پکیج مربع کنار پکیج مورد نظر سبز رنگ خواهد شد.

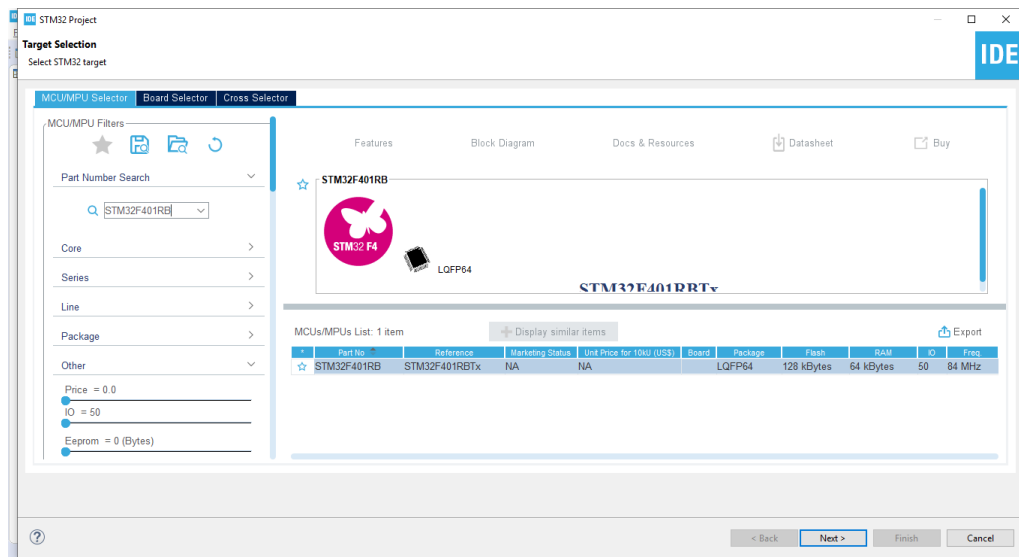
۳- از مسیر شکل زیر یک پروژه جدید بسازید



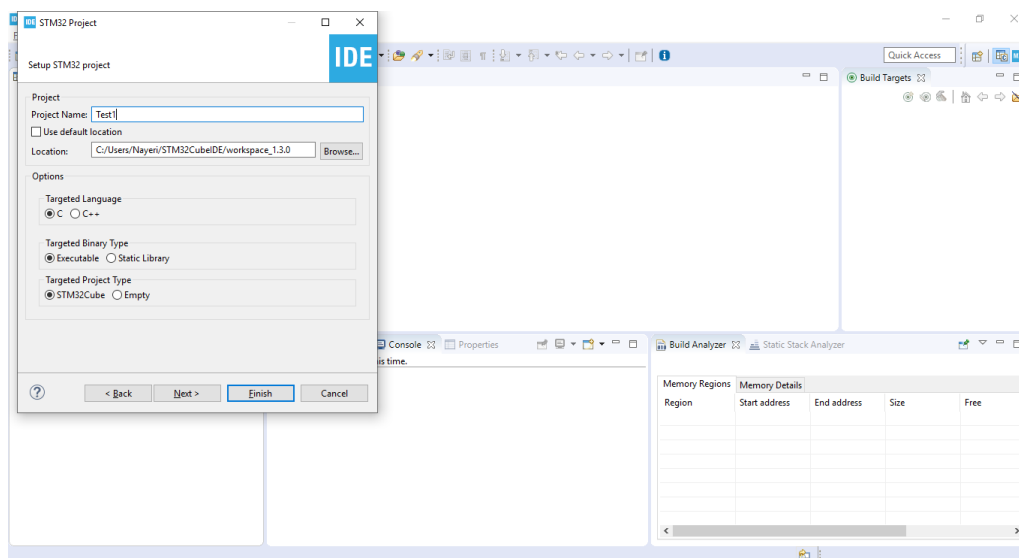
سپس صفحه ای به صورت شکل زیر باز خواهد شد. در قسمت Part Number Search میکرو کنترلر

مطابق جدول زیر را تایپ کرده در پنجره وسط آن را انتخاب کنید و بر روی Next کلیک کنید

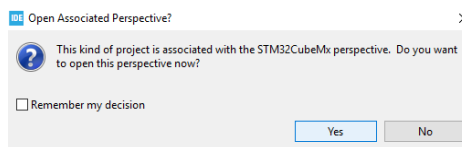
میکروکنترلر	باقیمانده شماره دانشجویی بر ۳
STM32F103C6	۰
STM32F103R6	۱
STM32F103T6	۲



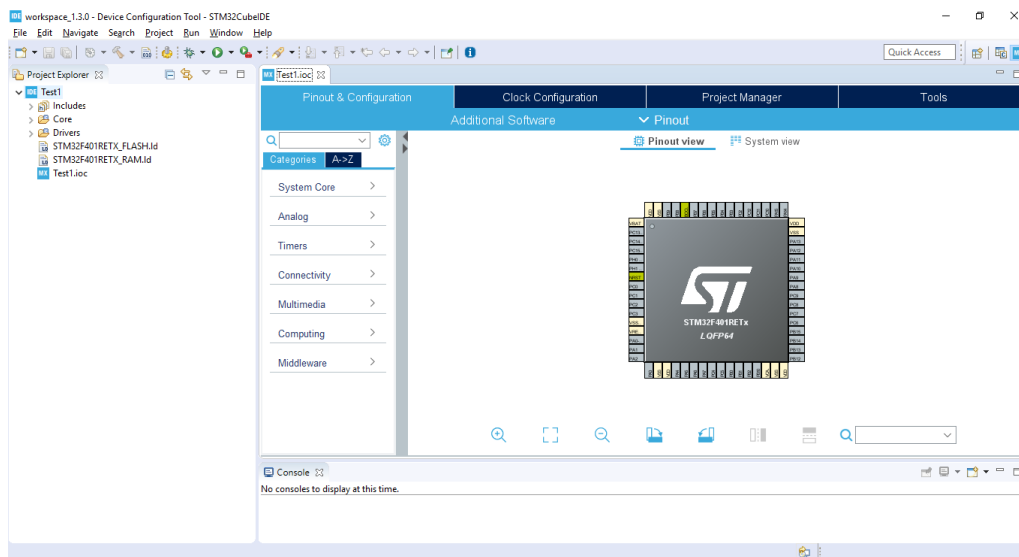
سپس پنجره ای به صورت شکل زیر باز خواهد شد که در آن میبایست یک نام برای پروژه انتخاب کرده و سپس مسیری را که پروژه در آن ذخیره می شود انتخاب کنید:



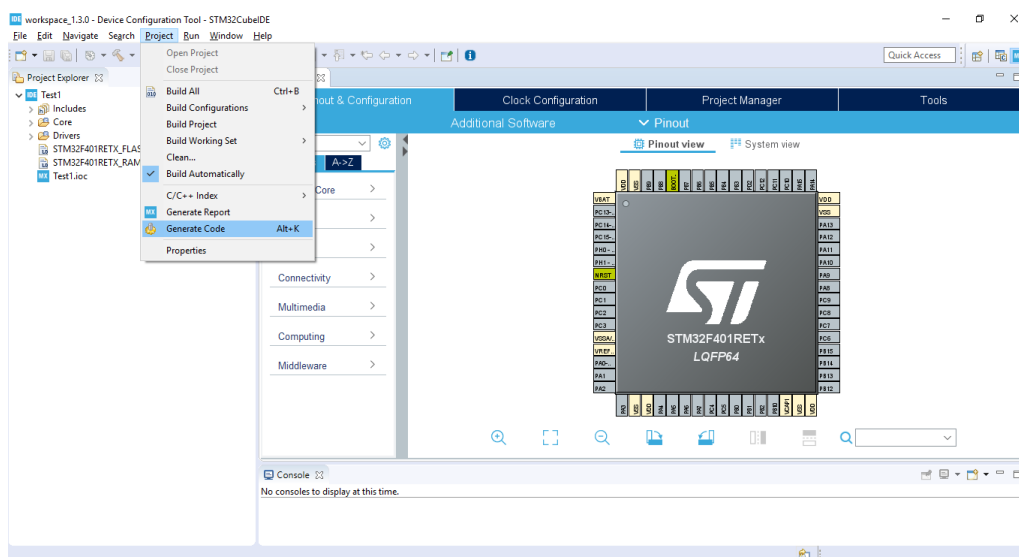
سپس بر روی گزینه Finish کلیک کنید. در صورتی که پیامی به صورت شکل زیر ظاهر شد، بر روی گزینه Yes کلیک کنید



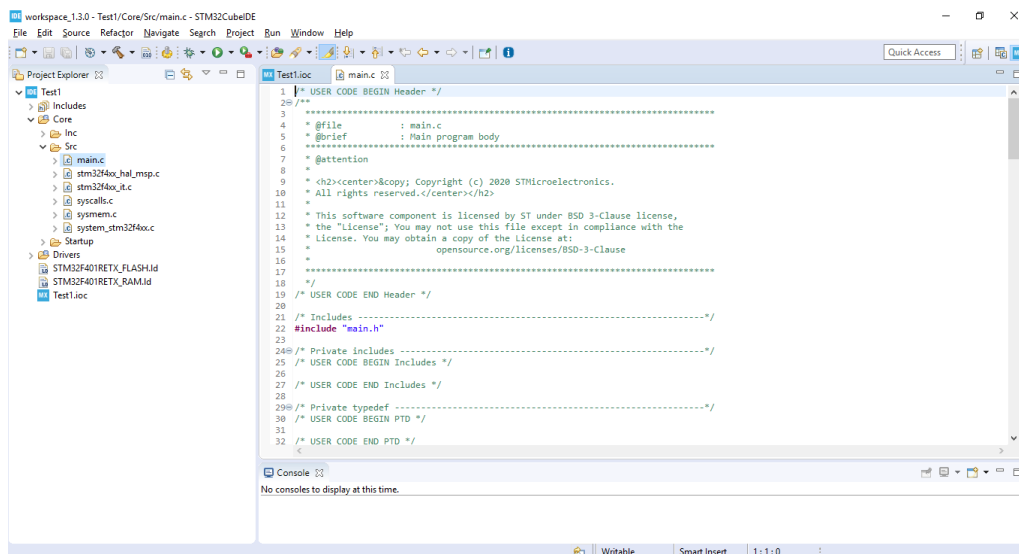
صفحه ای مطابق با شکل زیر ظاهر خواهد شد که شبیه به نرم افزار CubeMX است و به راحتی می توانید تنظیمات مربوط به بخش های مختلف میکروکنترلر را روی آن انجام دهید.



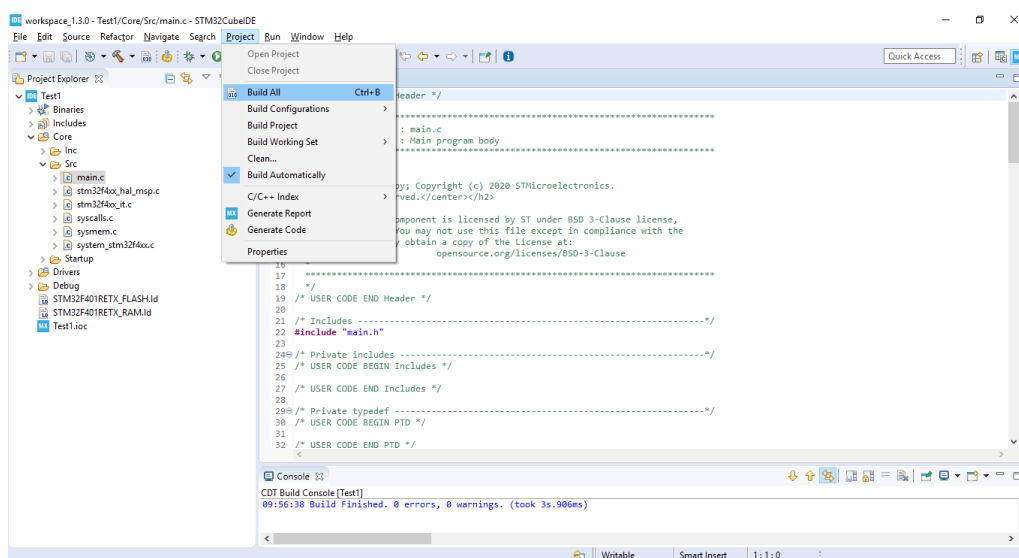
پس از انجام تنظیمات مورد نظر مطابق شکل زیر روی **Generate Code** کلیک کنید:



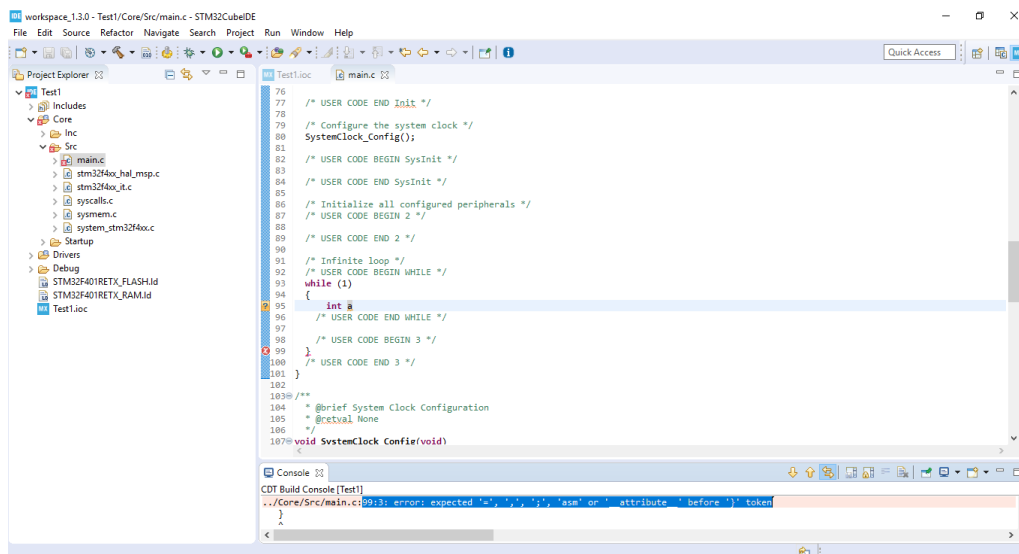
سپس مطابق شکل زیر از ستون سمت چپ فایل **main.c** را انتخاب کنید. صفحه ای مطابق شکل زیر باز خواهد شد که در واقع بدنه اصلی برنامه است و مانند نرم افزار Keil می توانید در آن کدنویسی کنید.



پس از اینکه کدهای مورد نظر را نوشتید می بایست کد مورد نظر را Build کنید تا از خطاها و هشدارهای احتمالی مطلع شوید. برای این منظور مطابق شکل زیر Build All را انتخاب کنید. همانطور که مشاهده می شود در Console پایین نرم افزار تعداد خطاها و هشدارها مشخص است.

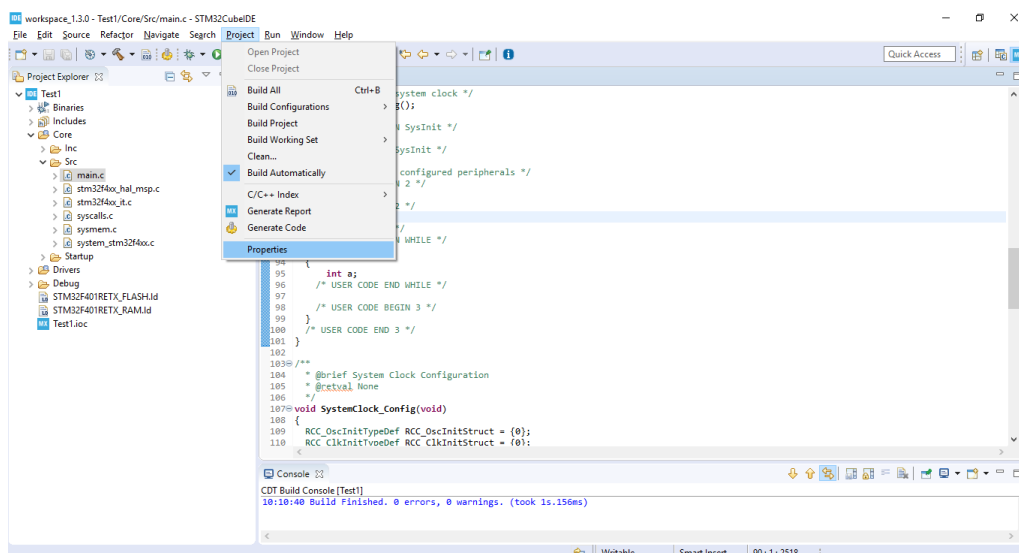


در صورتی که خطا یا هشدار در برنامه نویسی وجود داشته باشد، مطابق شکل زیر شماره خط و علت آن در Console قابل مشاهده است.

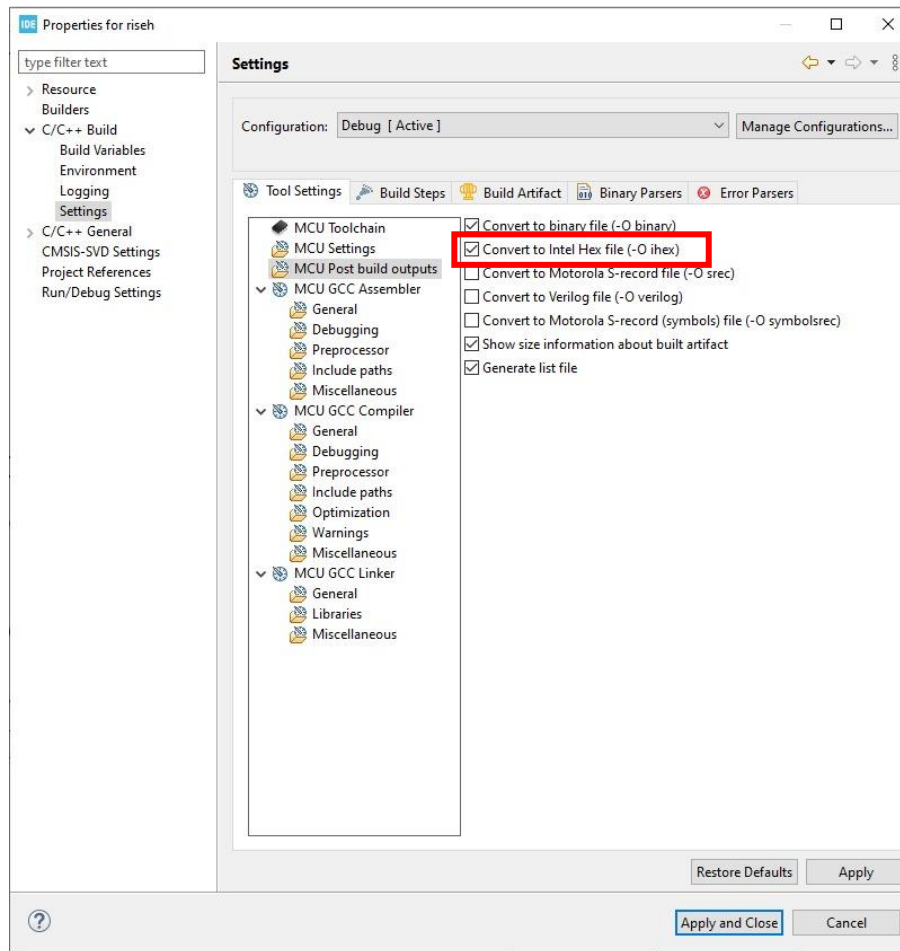


۴- ساخت فایل hex.* جهت استفاده در نرم افزار Proteus

مطابق شکل های زیر عمل کنید

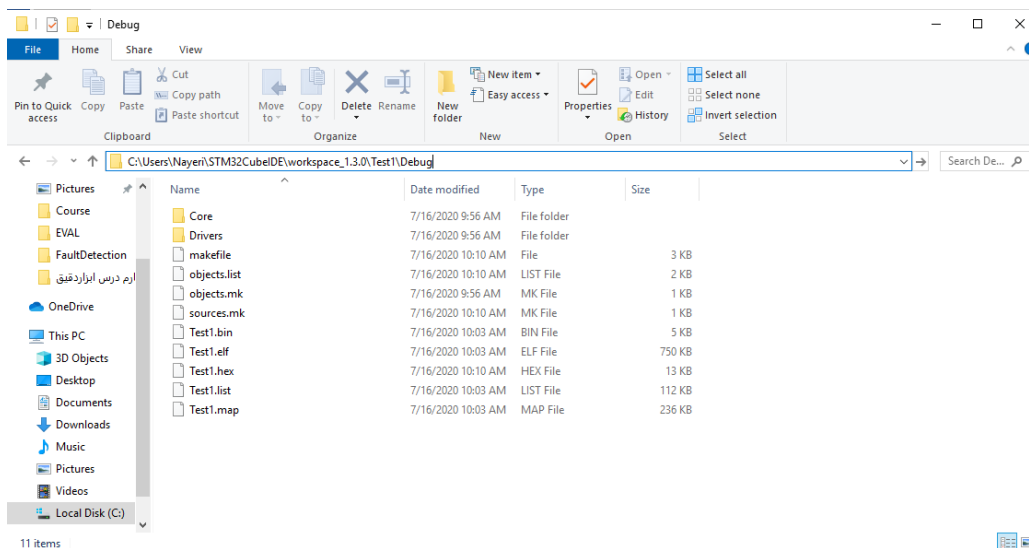


در بخش MCU Post-Build outputs تیک مربوط به Convert to intel Hex file و بر روی Apply کلیک کنید.



سپس پروژه را مجدداً Build All کنید.

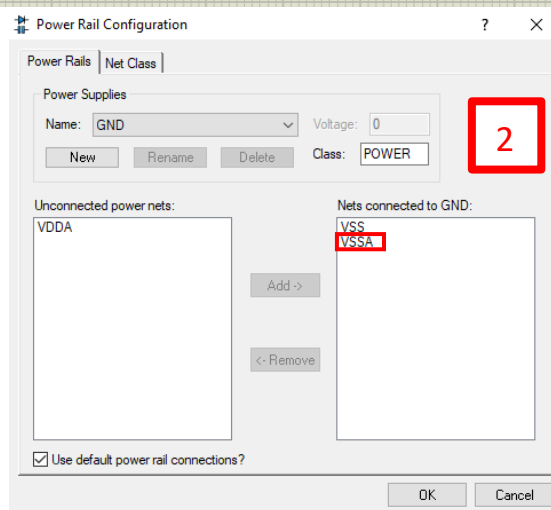
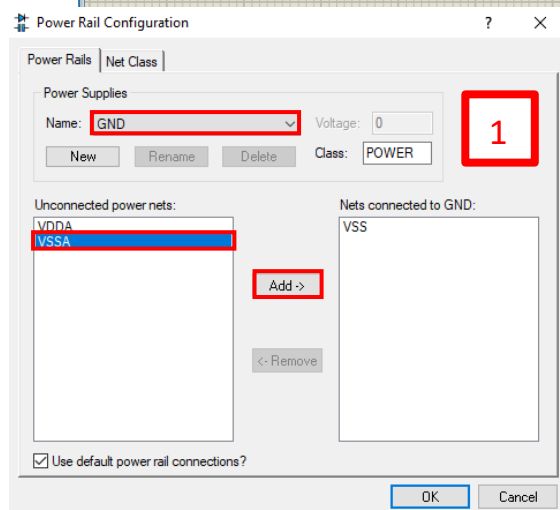
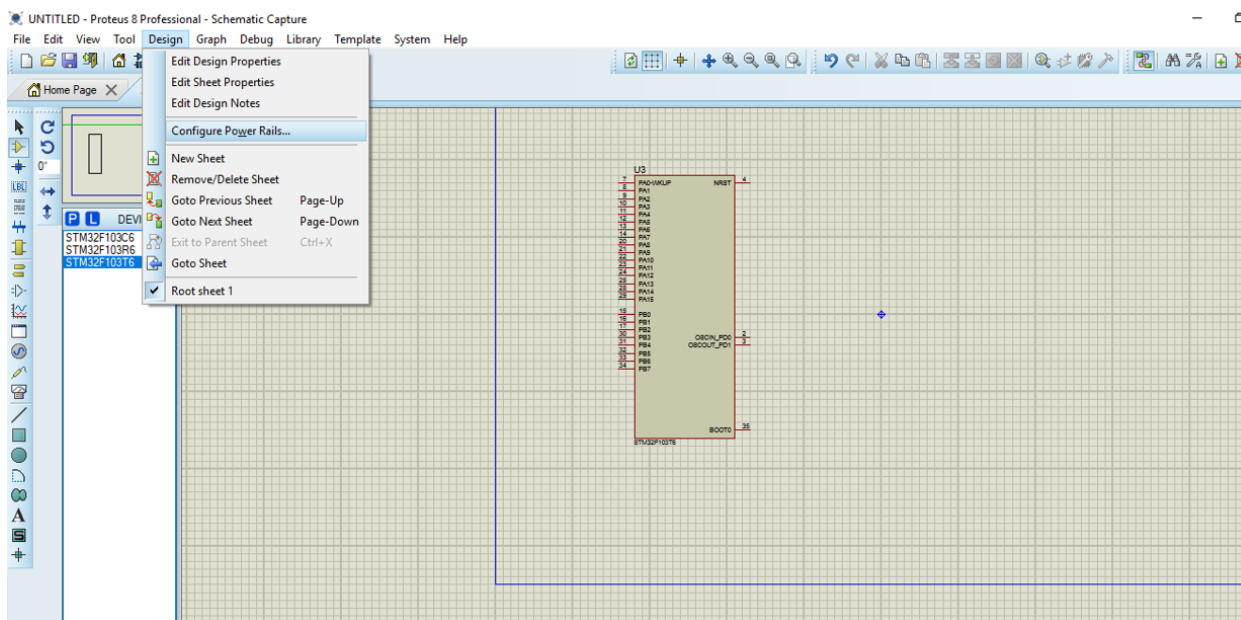
برای دسترسی به فایل *.hex به آدرس ساخت پروژه مطابق شکل زیر بروید:

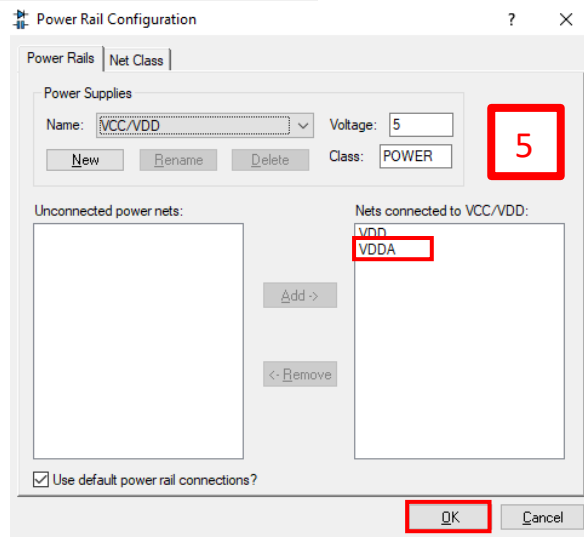
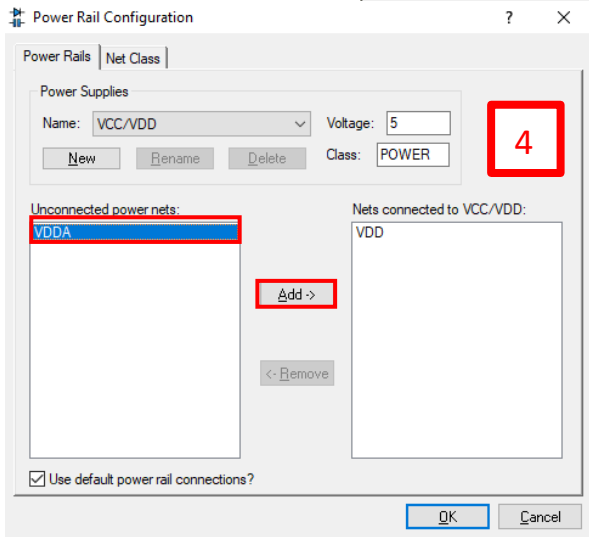
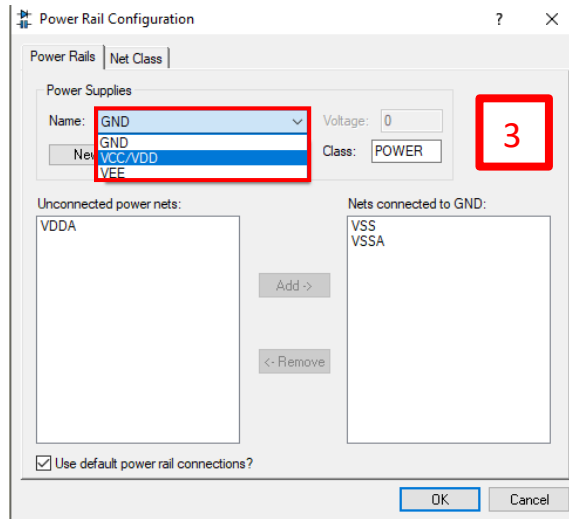


۵- نرم افزار Proteus را نصب کنید. (ورژن 8.9 به بالا را نصب کنید)

توجه: باتوجه به اینکه این نرم افزار کرک شده است و بسته به کامپیوتر شما و نوع کرک، ممکن است پس از مدتی که نرم افزار باز است، خود به خود بسته شود بنابراین پس از هر تغییر در محیط شبیه سازی بلافاصله پروژه را Save کنید تا تغییرات از دست نروند.

۶- برای شبیه سازی نیاز است که پایه های VDDA و VSSA را به ترتیب به تغذیه و زمین میکروکنترلر متصل کنید. برای این منظور مطابق شکل های زیر عمل کنید:





۷- مقدار کلاک اصلی میکرو کنترلر را به صورت زیر تنظیم کنید

$HCLK(MHz)$	باقیمانده شماره دانشجویی بر ۹
10	۰
15	۱
20	۲
26	۳
30	۴
36	۵
40	۶
44	۷
52	۸
60	۹

➤ روشن و خاموش کردن پایه دیجیتال با استفاده از ورودی های دیجیتال

- بر روی PORTA پایه های X1 و X2 را به عنوان خروجی دیجیتال به صورت زیر تعریف کنید.

باقیمانده شماره دانشجویی بر $X1 = 5$

باقیمانده شماره دانشجویی بر $X2 = 5 + 7$

- یک LED سبز و یک LED قرمز به پایه های فوق متصل کنید.

توجه: در اتصال LED به صورت سخت افزاری، برای روشن کردن LED نیاز است که حداکثر جریان قابل تحمل آن را بدانیم که این مقدار با مراجعه به دیتاشیت آن مشخص می شود. در صورتی که LED مستقیم به پایه میکروکنترلر متصل شود حداکثر جریان خروجی به آن متصل می گردد و احتمال سوختن LED و یا پایه خروجی دیجیتال میکرو کنترلر زیاد است. به منظور محدود کردن جریان یک مقاومت با توجه به حداکثر جریان قابل تحمل با LED سری می شود. برای مثال اگر حداکثر جریان قابل تحمل LED ۵ میلی آمپر باشد و بخواهیم آن را به میکروکنترلر ARM (که ولتاژ خروجی دیجیتال آن در حالت معمول ۳/۳ ولت است) متصل کنیم، از یک مقاومت ۱ کیلو اهم جهت محدود کردن جریان (به مقدار حداکثر ۳/۳ میلی آمپر) استفاده می شود. (این موضوع با فرض مقاومت ناچیز LED در نظر گرفته شده است).

- بر روی PORTB پایه های Y1 و Y2 را به عنوان ورودی دیجیتال به صورت زیر تعریف کنید.

باقیمانده شماره دانشجویی بر $Y1 = 3$

باقیمانده شماره دانشجویی بر $Y2 = 2 + 4$

- به پایه های B.Y1 و B.Y2 بخش قبل دو Push button که خروجی آن ها **Pull up** است را متصل کنید.
- برنامه ای بنویسید که LED سبز را ۵۰۰ میلی ثانیه روشن و ۵۰۰ میلی ثانیه خاموش کند. برای این منظور می بایست از دستور HAL_GPIO_WritePin و دستور HAL_Delay استفاده کنید. برنامه نوشته شده را با استفاده از پروتئوس شبیه سازی کنید. برنامه میکروکنترلر (فایل های main.c ، *.hex و *.ioc) و برنامه پروتئوس (فایل *.pdsprj) را در یک فولدر با نام Q1a ذخیره کنید.
- برنامه ای بنویسید که با فشردن Push button مربوط به پایه B.Y1 ، LED سبز روشن و LED قرمز خاموش و با فشردن Push button مربوط به پایه B.Y2 ، LED سبز خاموش و LED قرمز روشن شود. برای این منظور به دستور HAL_GPIO_ReadPin نیاز خواهید داشت. برنامه نوشته شده را با استفاده

از پروتئوس شبیه سازی کنید. برنامه های میکروکنترلر (فایل های main.c ، *.hex و *.ioc) و برنامه پروتئوس (فایل *.pdsprj) را در یک فولدر با نام Q1b ذخیره کنید.

➤ وقفه خارجی

- پایه B.Y1 را به عنوان ورودی دیجیتال تعریف کنید.
 - پایه های B.Y2 بخش قبل را به عنوان وقفه خارجی حساس به لبه بالارونده تعریف کنید.
 - به پایه های بخش قبل دو Push button که خروجی آنها **Pull up** است را متصل کنید.
 - برنامه ای بنویسید که سناریوی زیر پیاده سازی شود
- I. در حالت عادی LED ها روشن باشند.
- II. در صورت فعال شدن وقفه B.Y2 ، LED سبز خاموش و LED قرمز هر ۱۰۰ میلی ثانیه روشن و خاموش شود. برای این منظور از دستور HAL_GPIO_TogglePin استفاده کنید.
- III. با فشردن کلید متصل به B.Y2 برنامه به حالت عادی برگردد.

توجه: پس از فعال کردن وقفه در Cubemx ، از مسیر

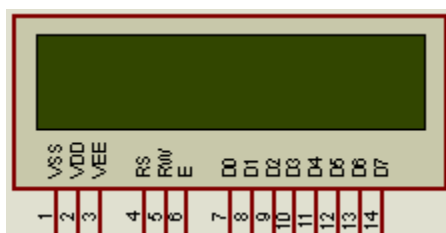
Core->Src->stm32f1xx_it.c

به روتین وقفه دسترسی خواهید داشت و می توانید کد های مورد نظر در هنگام فعال شدن وقفه را در این روتین بنویسید.

- برنامه نوشته شده را با استفاده از پروتئوس شبیه سازی کنید.
- برنامه های میکروکنترلر (فایل های main.c ، *.hex و *.ioc) و برنامه پروتئوس (فایل *.pdsprj) را در یک فولدر با نام Q2 ذخیره کنید.

➤ اتصال نمایشگر به میکروکنترلر

هدف از اجرای این بخش اتصال 16x2 Alphanumeric LCD به صورت شکل زیر به پورت B میکروکنترلر است. برای این نمایشگر کتابخانه های متنوعی تولید شده است.



- با جست و جو در اینترنت برنامه ای بنویسید که بتوان با استفاده از آن در خط اول نام و در خط دوم شماره دانشجویی شما را نمایش دهد.
- برنامه نوشته شده را با استفاده از پروتئوس شبیه سازی کنید. (توجه: برای راه اندازی LCD در پروتئوس پایه مربوط به RW را به زمین متصل کنید)
- برنامه های میکروکنترلر (فایل های main.c ، *.hex و *.ioc) و برنامه پروتئوس (فایل *.pdsprj) را در یک فولدر با نام Q3a ذخیره کنید.
- یک پروژه برنامه نویسی جدید ایجاد کنید. ۳ رقم سمت راست شماره دانشجویی خود را در متغیر a ذخیره کنید. سپس متغیر a را بر عدد ۷ تقسیم کرده و در متغیر b ذخیره کنید. سپس مقدار متغیر a را در خط اول و مقدار متغیر b را با ۴ رقم اعشار در خط دوم نمایش دهید. (برای این منظور می بایست از دستور sprintf استفاده کنید)

توجه: چنانچه دستور مورد نظر در cubeide کار نکرد راه حل را با جست و جو در اینترنت بیابید.

- برنامه های میکروکنترلر (فایل های main.c ، *.hex و *.ioc) و برنامه پروتئوس (فایل *.pdsprj) را در یک فولدر با نام Q3b ذخیره کنید.

➤ تایمر

- برنامه ای بنویسید که با استفاده از یک تایمر یک متغیر از مقدار اولیه ۰ هر ۵۰ میلی ثانیه یکبار، ۱ واحد به مقدار آن افزوده شود و هنگامی که به مقدار ۵۰ رسید، ریست شود. مقدار این متغیر را بر روی نمایشگر نشان دهید.

توجه: زمان شبیه سازی پروتئوس بر اساس زمان واقعی نیست بنابراین برای بررسی صحت عملکرد برنامه نوشته شده می بایست Animating Time را در نظر بگیرید.

- برنامه های میکروکنترلر (فایل های main.c ، *.hex و *.ioc) و برنامه پروتئوس (فایل *.pdsprj) را در یک فولدر با نام Q4 ذخیره کنید.

➤ مبدل آنالوگ به دیجیتال

در این بخش می خواهیم مبدل آنالوگ به دیجیتال میکروکنترلر ARM را بررسی و راه اندازی کنیم

با توجه به مینی پروژه قبل یک نمایشگر به میکروکنترلر متصل کنید سپس با مشاهده ویدئوی با لینک زیر برنامه ای بنویسید که سناریو زیر را پیاده سازی کند:

<https://www.youtube.com/watch?v=VaAl9hnPGiA>

سناریو:

فرص کنید یک ولتاژ دلخواه به میکروکنترلر متصل است و مبدل آنالوگ به دیجیتال از این ولتاژ نمونه برداری کرده و روی خط اول نمایشگر مقدار باینری خروجی مبدل آنالوگ به دیجیتال نمایش داده شده و بر روی خط دوم مقدار ولتاژ بر حسب ولت نمایش داده شود. ولتاژ را چندین بار تغییر دهید و از صحت عملکرد مبدل اطمینان حاصل کنید.

توجه: در پنجره Configure Power Rails در نرم افزار پروتئوس مقدار ولتاژ Power را روی ۳ ولت تنظیم کنید. و رزولوشن مبدل حتما روی ۱۲ بیت باشد.

- برنامه های میکروکنترلر (فایل های main.c ، *.hex و *.ioc و فایل کتابخانه های مورد استفاده) و برنامه پروتئوس (فایل *.pdsprj) را در یک فولدر با نام Q5 ذخیره کنید.

➤ ساخت ترازو

۱- قطعه Loadcell را از کتابخانه نرم افزار پروتئوس به محیط شبیه سازی اضافه کنید. با کلیک راست بر روی این قطعه و انتخاب گزینه Edit Properties پنجره ای مطابق شکل زیر باز خواهد شد. تنظیمات را مطابق شکل انجام داده سپس بر روی OK کلیک کنید.

$$a = 0.05 \times (\text{mod}(SN, 4) + 1)$$

↓
0.15

Edit Component

Part Reference: LC1

Part Value: LOADCELL

Element: [] New

LISA Model File: LOADCELL

% Step: 1

Full Scale (%): 100

Bridge Resistance (Ohms): 350

Sensitivity (mV/V): a

Advanced Properties:

Output Offset Voltage: 1m

Other Properties:

☐ Exclude from Simulation
☐ Exclude from PCB Layout
☐ Exclude from Current Variant

☐ Attach hierarchy module
☐ Hide common pins
☐ Edit all properties as text

Hidden: ☐ OK Cancel

۲- توضیح دهید پارامتر های Full Scale و Sensitivity در Loadcell به چه معنی هستند. (در گزارش آورده شود).

۳- با استفاده از پروتئوس آزمایشی طراحی کنید که به وسیله آن بتوانید مشخص کنید مقاومت هر ساق پل چقدر است. نام فایل را res.pdsprj بگذارید و با فایل های پروژه ارسال کنید.

۴- با فرض اینکه مقدار وزن Full Scale برابر با ۵۰۰ گرم است و به این نیروسنج می توان ولتاژ ۵ تا ۱۲ ولت اعمال کرد و با استفاده از تقویت کننده ابزار دقیق AD623 مداری طراحی کنید که وزن بین ۰ تا ۵۰۰ گرم را به ولتاژ بین ۰ تا x ولت نگاشت کند، سپس جدول زیر را تکمیل کنید: (در گزارش آورده شود)

$x = \text{mod}(SN, 3) + 1 + 0.1 \times \text{mod}(SN, 9)$

W(g)	Vout (Amplifier) (v)
10	
20	

30	
...	
90	
100	
200	
300	
400	
500	

- ۵- با استفاده از داده های بدست آمده در بخش قبل و جعبه ابزار cftool نرم افزار MATLAB بهترین خط ممکن را بر داده ها برازش کرده و معادله خط و خطای برازش را بدست آورید. (در گزارش آورده شود)
- ۶- با استفاده از معادله بدست آمده از بخش قبل برنامه ای بنویسید که وزن را بر کیلوگرم اندازه گیری کرده و با ۲ رقم اعشار بر روی نمایشگر نشان دهد.
- ۷- برنامه های میکروکنترلر (فایل های main.c ، *.hex و *.ioc و فایل کتابخانه های مورد استفاده) و برنامه پروتئوس (فایل *.pdsprj) را در یک فولدر با نام Q6 ذخیره کنید.
- ۸- دقت و رزولوشن سیستم اندازه گیری وزنی که طراحی کرده اید، را به صورت تئوری محاسبه کنید و تحلیل کنید که این دو پارامتر به حسگر محدود شده اند یا به میکروکنترلر؟ (در گزارش آورده شود)

➤ راه اندازی حسگر دما LM50

- ۱- با توجه به دیتاشیت حسگر، پارامترهای محدوده اندازه گیری، حساسیت، دقت و پایه های تغذیه و خروجی را مشخص کنید. (در گزارش آورده شود)
- ۲- برنامه ای بنویسید که دمای ۲۰- تا ۶۰ درجه سانتی گراد را با بالاترین دقت ممکن اندازه گیری کرده و با دو رقم اعشار بر روی نمایشگر نشان دهد. برنامه های میکروکنترلر (فایل های main.c ، *.hex و *.ioc و فایل کتابخانه های مورد استفاده) و برنامه پروتئوس (فایل *.pdsprj) را در یک فولدر با نام Q7 ذخیره کنید.
- ۳- دقت و رزولوشن سیستم اندازه گیری دمایی که طراحی کرده اید، چقدر است و تحلیل کنید که این دو پارامتر به حسگر محدود شده اند یا به میکروکنترلر؟ (در گزارش آورده شود)

➤ راه اندازی حسگر اندازه گیری فاصله GP2Y0A700K0F

۱- حسگر را با منبع تغذیه ۵ ولت راه اندازی کرده و با متصل کردن یک ولت‌متر به خروجی حسگر جدول

زیر را تکمیل کنید: (در گزارش آورده شود)

d(cm)	Vout(v)
100	
120	
140	
160	
...	
300	
320	

۴- با استفاده از داده های بدست آمده در بخش قبل و جعبه ابزار cftool نرم افزار MATLAB بهترین

منحنی مشخصه ممکن را بر داده ها برازش کرده و معادله آن را بدست آورید. (در گزارش آورده شود)

۵- با استفاده از معادله بدست آمده از بخش قبل و استفاده از میکروکنترلر برنامه ای بنویسید که فاصله را

بر حسب سانتی متر اندازه گیری کرده و با ۲ رقم اعشار بر روی نمایشگر نشان دهد. برنامه های

میکروکنترلر (فایل های main.c ، *.hex و *.ioc و فایل کتابخانه های مورد استفاده) و برنامه

پروتئوس (فایل *.pdsprj) را در یک فولدر با نام Q8 ذخیره کنید.

۲- دقت و رزولوشن سیستم اندازه گیری فاصله ای که طراحی کرده اید، چقدر است و تحلیل کنید که این

دو پارامتر به حسگر محدود شده اند یا به میکروکنترلر؟ (در گزارش آورده شود)

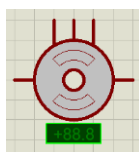
➤ راه اندازی انکودر افزایشی

✓ راه اندازی موتور



۱- از کتابخانه نرم افزار پروتئوس MOTOR-ENCODER را به بخش شبیه سازی اضافه کنید و پارامتر زیر

را تنظیم کنید:

$$\text{Pulses per Revolution} = 260 + \text{دو رقم راست شماره دانشجویی}$$



۲- کتابخانه پروتئوس دارای یک کلید فشاری با نام BUTTON است که میتوانید به صورت Normally Open و یا Normally Close به صورت شکل های زیر از آن استفاده کنید

	
Normally Close	Normally Open

توجه کنید که برای تغییر مود از Normally Open به Normally Close کافیست روی دایره بالا سمت راست کلید فشاری کلیک کنید.

۳- با استفاده از یک منبع تغذیه ۱۲ ولت، تعدادی رله دو کنتاکت با نام RELAY2P و تعدادی کلید فشاری مداری طراحی کنید که دارای ۳ کلید فشاری بوده و به صورت زیر عمل می کند:

I. کلید فشاری ۱ بوبین یک رله که وظیفه رساندن ولتاژ به دو سر موتور را دارد بر عهده دارد را تحریک می کند. در صورتی که کلید فشاری ۱ فشرده باقی بماند ولتاژ به موتور می رسد و موتور می چرخد.

II. با یکبار فشردن کلید فشاری ۲ موتور در جهت عقربه های ساعت (CW) می چرخد. توجه کنید که نباید کلید فشاری ۲ را نگه دارید و تنها با یکبار فشردن می بایست حالت مد نظر ایجاد شود. (راهنمایی: نیاز به مدار خودنگه دار برای رله است)

III. با یکبار فشردن کلید فشاری ۳ موتور در خلاف جهت عقربه های ساعت (CCW) میچرخد. توجه کنید که نباید کلید فشاری ۳ را نگه دارید و تنها با یکبار فشردن می بایست حالت مد نظر ایجاد شود.

۴- برنامه پروتئوس (فایل *.pdsprj) را در یک فولدر با نام Q9a ذخیره کنید.

✓ اندازه گیری سرعت

۱- در این بخش از مدار قبلی که طراحی کرده اید استفاده کنید. برای این منظور ولتاژ منبع تغذیه را روی ۱۲ ولت تنظیم کرده، کلید فشاری ۱ را در حال بسته نگه دارید و پالس های A و B آنکودر را توسط اسیلوسکوپ بر روی یکدیگر مشاهده کنید. به تغییرات پالس های A و B وقتی جهت چرخش موتور به واسطه کلیدهای فشاری ۲ و ۳ تغییر می کند توجه کنید.

۲- پالس A را به میکروکنترلر متصل کرده، برنامه ای بنویسید که سرعت موتور را بر حسب RPM در خط اول با دو رقم اعشار نمایشگر نشان دهد.

- ۳- پالس B را به میکرو کنترلر متصل کرده برنامه ای بنویسید که در خط دوم سرعت جهت چرخش موتور (CW یا CCW) را نمایش دهد.
- ۴- دقت اندازه گیری سرعت، حداقل سرعت و حداکثر سرعت قابل اندازه گیری را با تنظیماتی که انجام داده اید را مشخص کنید. (در گزارش آورده شود)
- ۵- برنامه پروتئوس (فایل *.pdsprj) و برنامه میکروکنترلر (فایل های main.c ، *.hex و *.ioc و فایل کتابخانه های مورد استفاده) را در یک فولدر با نام Q9b ذخیره کنید.

لطفا در ارسال به موارد زیر توجه بفرمایید ، در صورت عدم رعایت هر یک از موارد زیر تمرین شما تصحیح نخواهد شد :

- تنها به پروژه هایی که با STM32CubeIDE انجام شده اند نمره تعلق خواهد گرفت.
- فولدرهای مربوط به برنامه ها را به صورت یک فایل zip تجمیع و با نام student_number.zip ارسال شوند .
- به تمرین هایی که به صورت مشابه حل شده اند نمره ای تعلق نخواهد گرفت.

همواره موفق باشید