آزمایش 3

طاها موسوی 98243058

نیلوفر مرادی جم 97243063

گروه 2

# **سوالات تحلیلی:**

**1- انواع وقفه را نام برده و با یکدیگر مقایسه کنید. اگر اتفاق افتادن چندین درخواست وقفه دارای اشتراکی زمانی باشد، عملکرد سیستم در M4-Cortex و قبل از آن را توضیح داده و با هم مقایسه کنید. اگر دو درخواست وقفه همزمان رخ دهند، سیستم به چه صورت عمل خواهد کرد؟**

1. نوع وقفه (interrupt) داریم:

**اول)** نوع اول interrupt requests است که آسنکرون می‌باشد و به کدی که در آن زمان اجرا می‌شود ارتباط ندارد. در این نوع وقفه یک سیگنال به پروسسور ارسال می شود تا فوری فرآیند فعلی متوقف شود و روال interrupt handler اجرا شود.

**دوم)** نوع دوم non-maskable interrupt است که همانند وقفه نوع اول می‌باشد با این تفاوت که قابل چشم پوشی نیست. این نوع وقفه معمولا در شرایطی ایجاد می‌شود که یک مشکل غیر قابل حل در سیستم هست.

**سوم)** نوع سوم Exceprtion ها و مشکلاتیند که توسط هسته پردازنده ایجاد شدند. این وقفه‌ها مربوط به اجرا شدن اینستراکشن های خاصیند همانند مثلا وقتی overflow رخ می‌دهد.

**چهارم)** نوع چهارم آن نیز systick timer است که وقفه هایی با زمان اجرای مشخص ایجاد می‌کند. سورس کلاک آن مانند cortex-m cpu است.

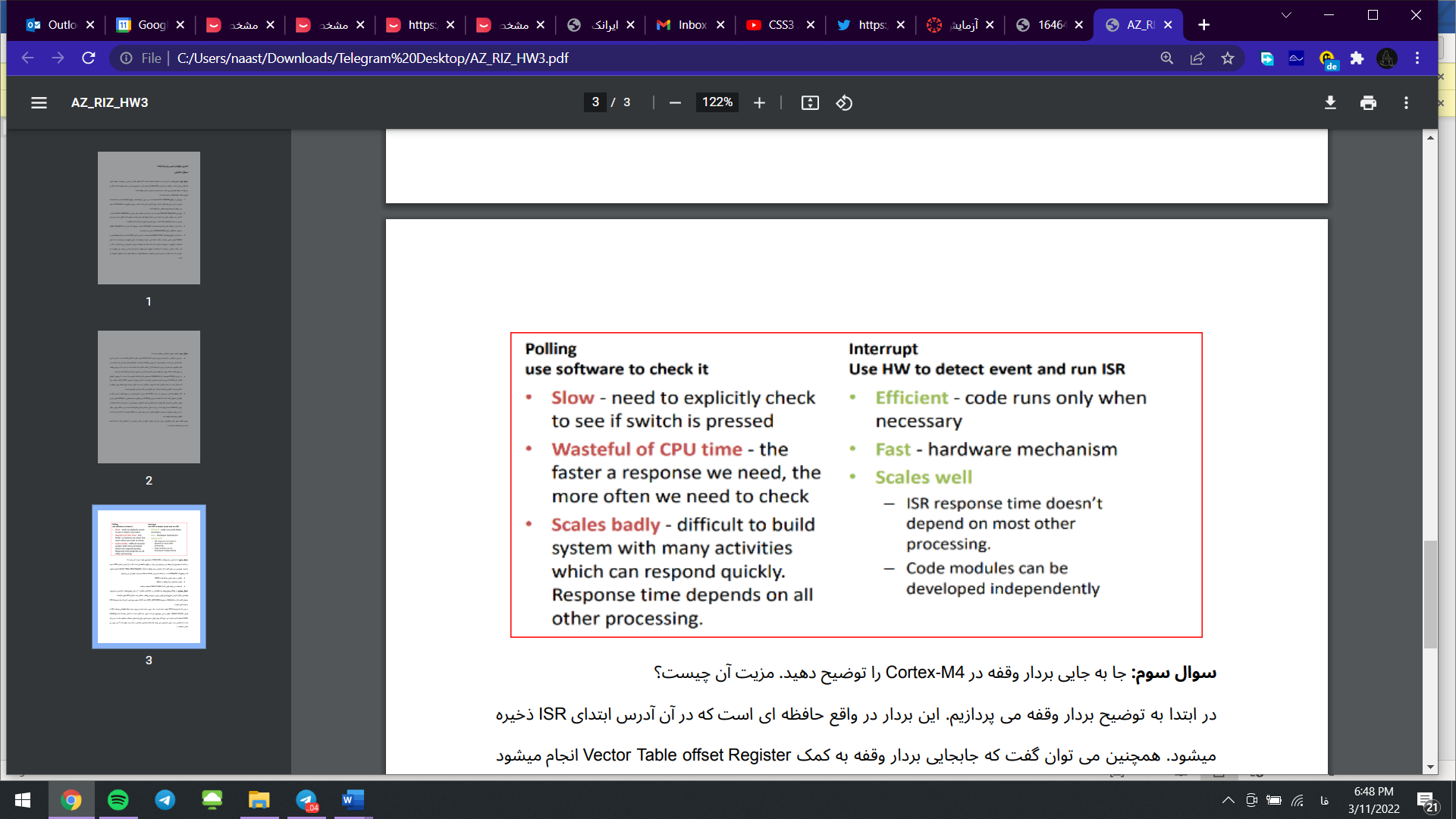
وقفه هایی متفاوت priority متفاوت دارند که در صورتی که همزمان رخ دهند، پردازنده هر کدام که عدد priorityکوچکتری داشته باشد را در الویت قرار می‌دهد.

1. **- تفاوت روش سرکشی و وقفه چیست؟**

**اول)** سرعت: در سرکشی باید دقیق چک کنیم فرآیند موردنظر اتفاق افتاده یا نه و بنابرین سرعت کمتری دارد اما در وققه فرآیند سخت‌افزاری است پس سرعتش بیشتر است.

**دوم)** بهره زمانی و بهره وری: در سرکشی هر چه پاسخ سریع‌تری بخواهیم، باید بیشتر چکش کنیم و بنابرین وقت زیادی می‌گیرد اما وقفه به دلیل سرعت بالا، بهره وری زمانی بهتری دارد. همچنین برخلاف سرکشی، کد وقفه تنها در زمان مورد نیاز اجرا می‌شود پس اجرای تکراری نداریم.

**سوم)** هزینه توسعه: اگر در یک فرآیند بزرگ وقفه بخواهیم، بدلیل وابستگی اجرای وقفه به بسیاری از فرآیند های دیگر، امکان توسعه چند بخش به صورت همزمان وجود ندارد و در مقیاس بزرگ به کار بسیار زیادی نیازمندیم. اما در وققه هر بخش از کد به صورت غیر وابسته به بقیه اجزای کد اجرا می شود و امکان توسعه همزمان بخش های مختلف وجود دارد و هزینه توسعه آن نیز کاهش می یابد.



**3- جا به جایی بردار وقفه در M4-Cortex را توضیح دهید. مزیت آن چیست؟**

بردار وقفه در واقع حافظه‌ای می‌باشد که آدرس ابتدای ISR در آن ذخیره می‌شود. جابه‌جایی بردار وقفه نیز با یک رجیستر قابل برنامه‌ریزی با نام Vector Table Offset Register انجام می‌شود.

موارد استفاده :

* برنامه هایی که boot loader دارند
* ذخیره کردن برنامه ها در رم
* تغییر داینامیک بردار وقفه در حافظه ای

**4 - در هنگام وقوع وقفه چه اطالعاتی در LR قرار میگیرد؟ از زمان وقوع وقفه )با فرض عدم وجود وقفه ای دیگر( تا زمان شروع اجرای اولین روتین سرویس وقفه، حداکثر چند سیکل CPU طور میکشد؟**

در هنگام وقفه به همراه LR ، EXC\_RETURN کد ذخیره شده که CPU آن را تولید می‌کند. در این کد اطلاعاتی همانند این که رجیسترها از کدام STACK POINTER(SP) باید خوانده شوند و این که به کدام مد باید برگردیم ( مثلا هنگام مد هندلر یا نخ در حال اجرا(ترد)) و با بیت های 4 تا 7 نیز این که از واحد FLOATING POINT استفاده کرده ایم یا نه، ذخیره می شود.

تنها 12 سایکل از زمان درخواست exception تا اجرای اولین اینستراکشن را می‌دهند.

**رفرنس های سوالات تحلیلی**:  
- کلاس درس و اسلاید های درسی

# دستور کار:

طبق صورت سوال باید یک ماشین حساب طراحی میکردیم.

موارد مورد استفاده:

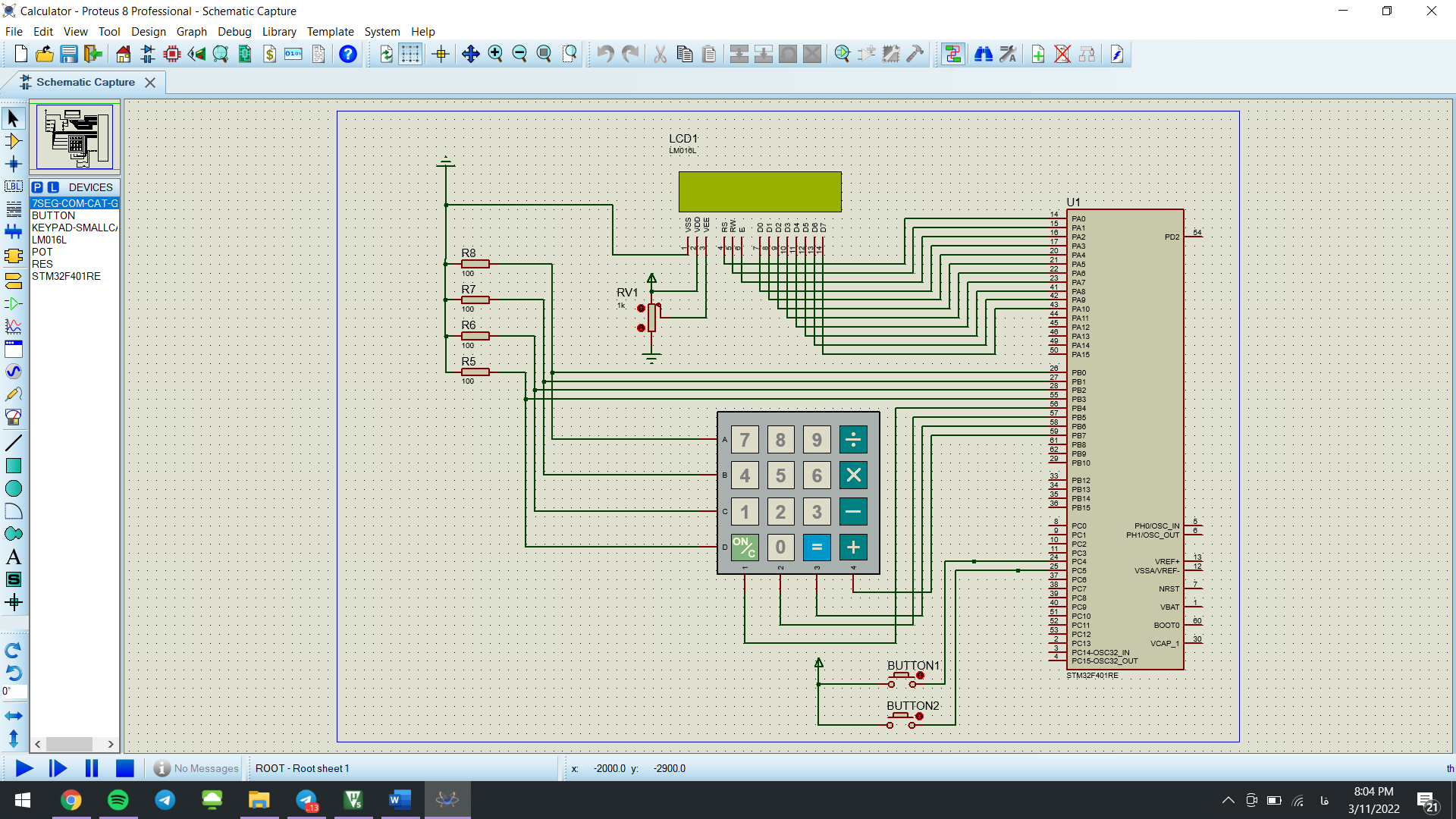
Lcd

Keypad

Stm32f401 microcontroller

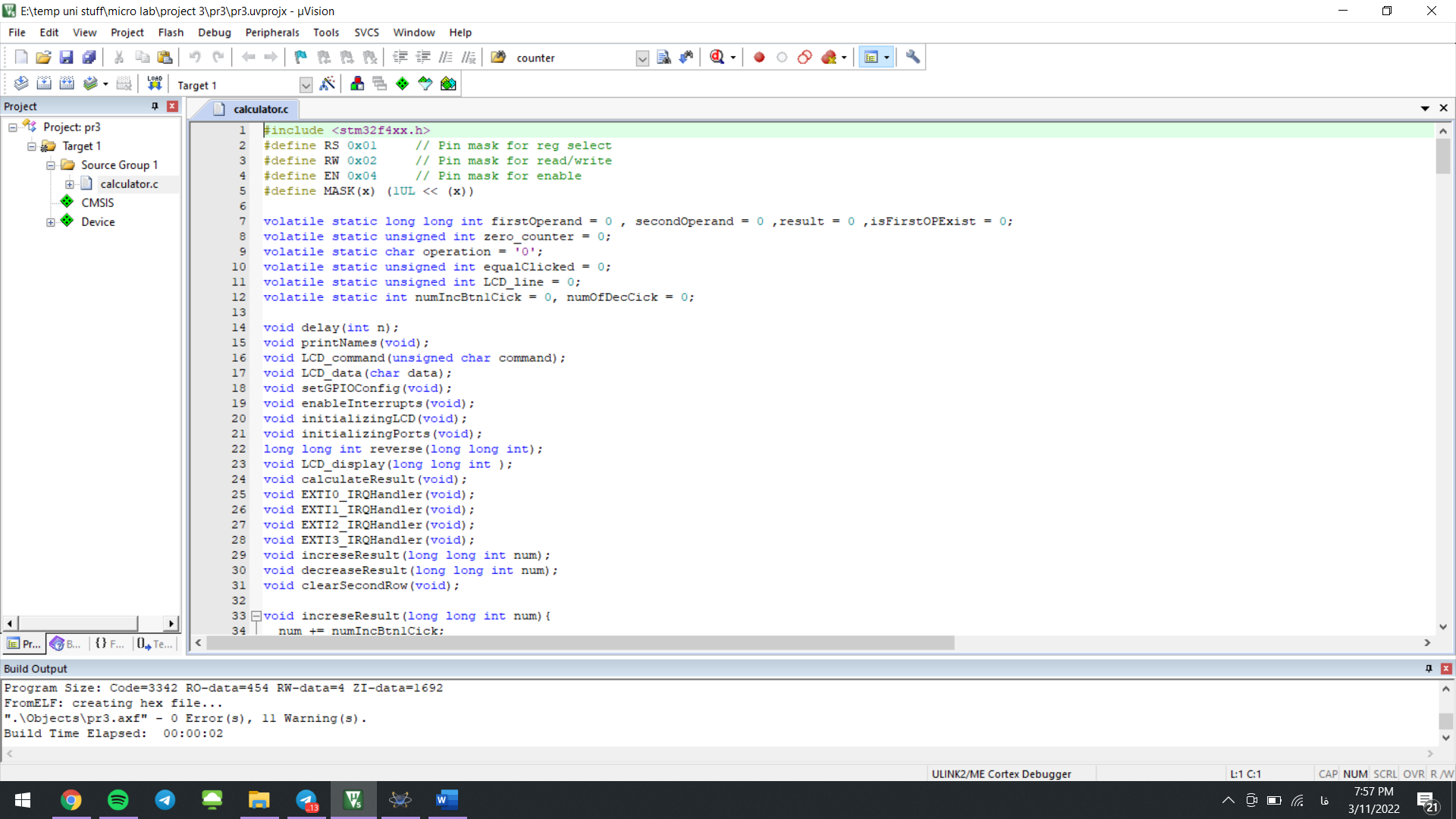
buttons

برای این بخش از قالب کد ارائه شده در گروه درس استفاده کردیم و یکسری روتین های موردنظرمان را به آن اضافه کردیم. پین های 0 تا 10 در GPIOA را به LED تخصیص داده ایم. GPIOB هم مختص keypad می باشد که چهارپین اول ورودی به میکرو و چهارپین دوم را خرو جی از میکرو قرار داده ایم.



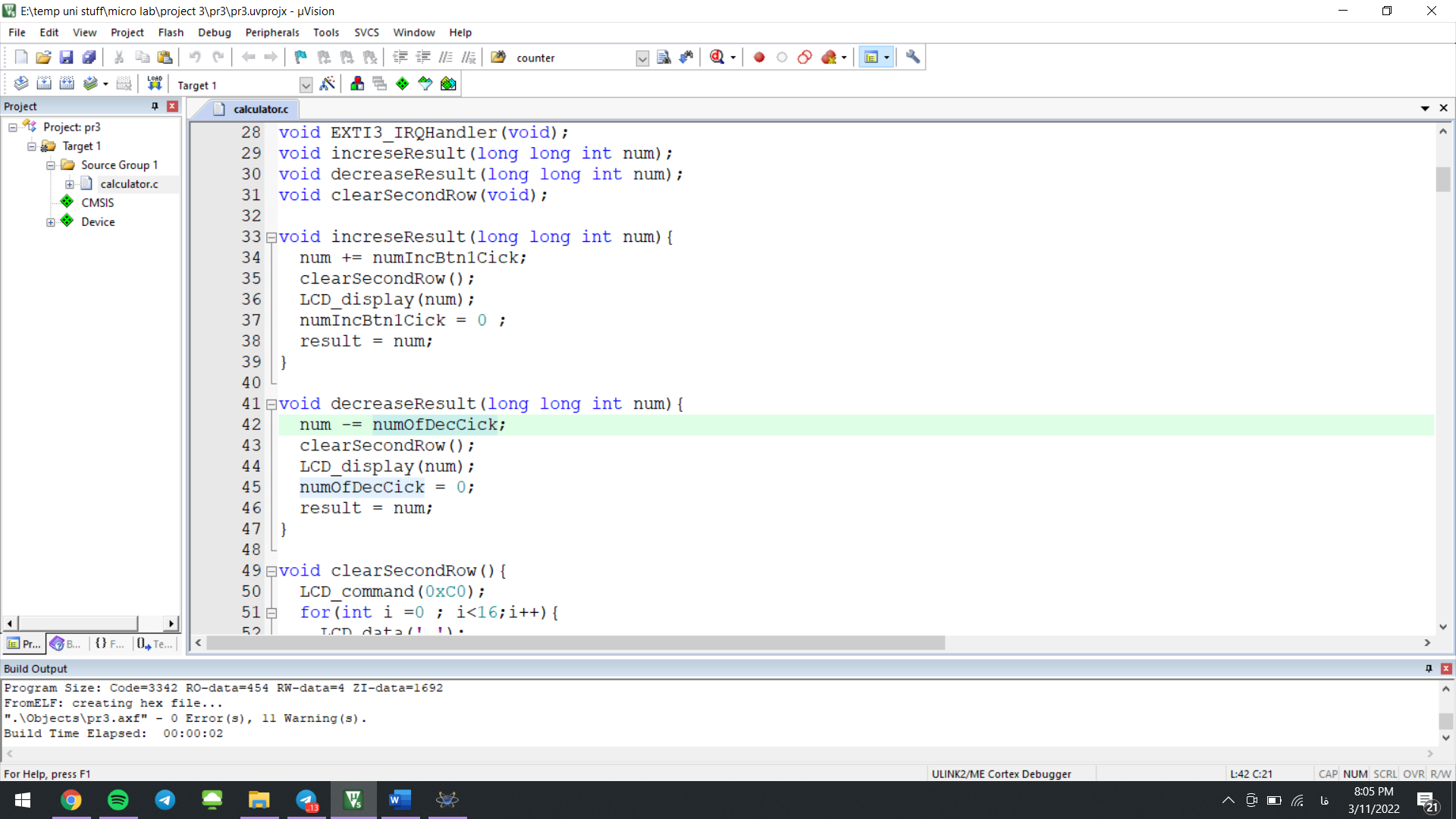
در ادامه لازم است کد c برای میکروکنترلر را طراحی کنیم و فایل هگز آن را برای اجرای برنامه بدهیم.

تعریف متغیرها و توابع:

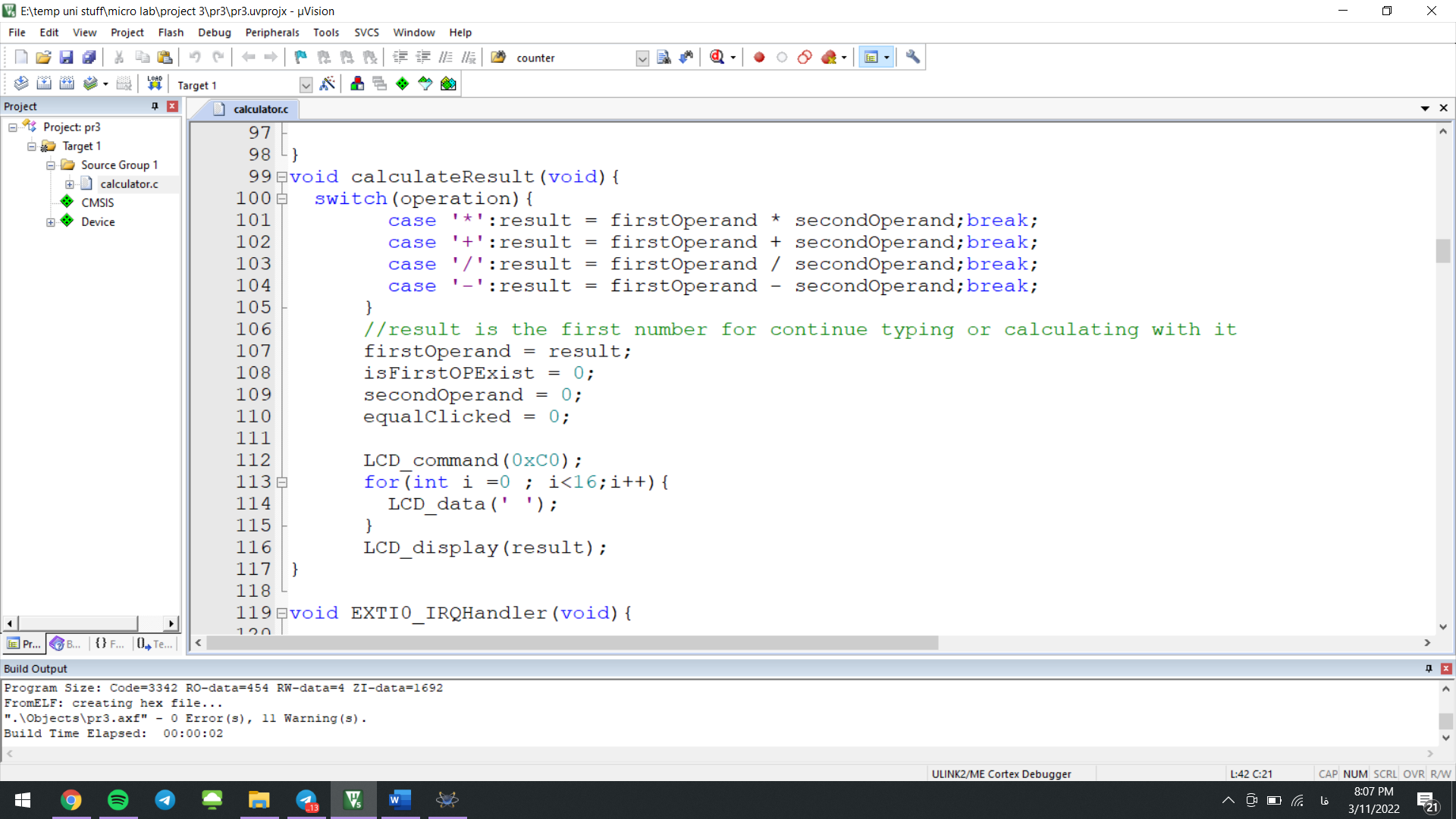


FirstOperand و secondOperand دو عملوند ما در عملیات ها هستند. وقتی که یک اپراتور را وارد می کنیم متغیر isFirstOpExist یک میشود و می فهمیم شماره دوم باید وارد شود. متغیر equalClicked اگر یک بشود باید نتیجه را چاپ کنیم.

توابع افزایش و کاهش نتیجه:

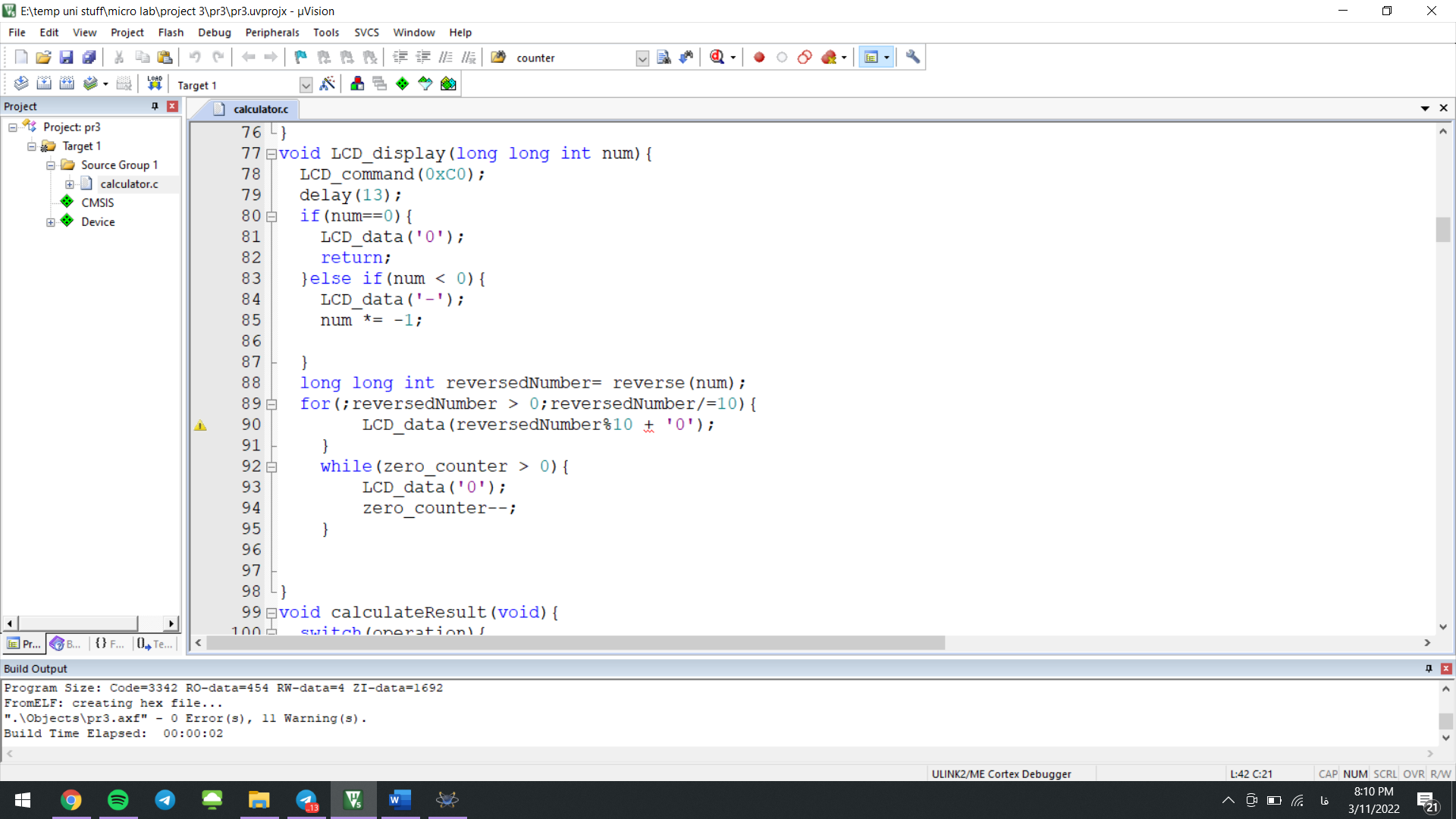


تابع محاسبه:



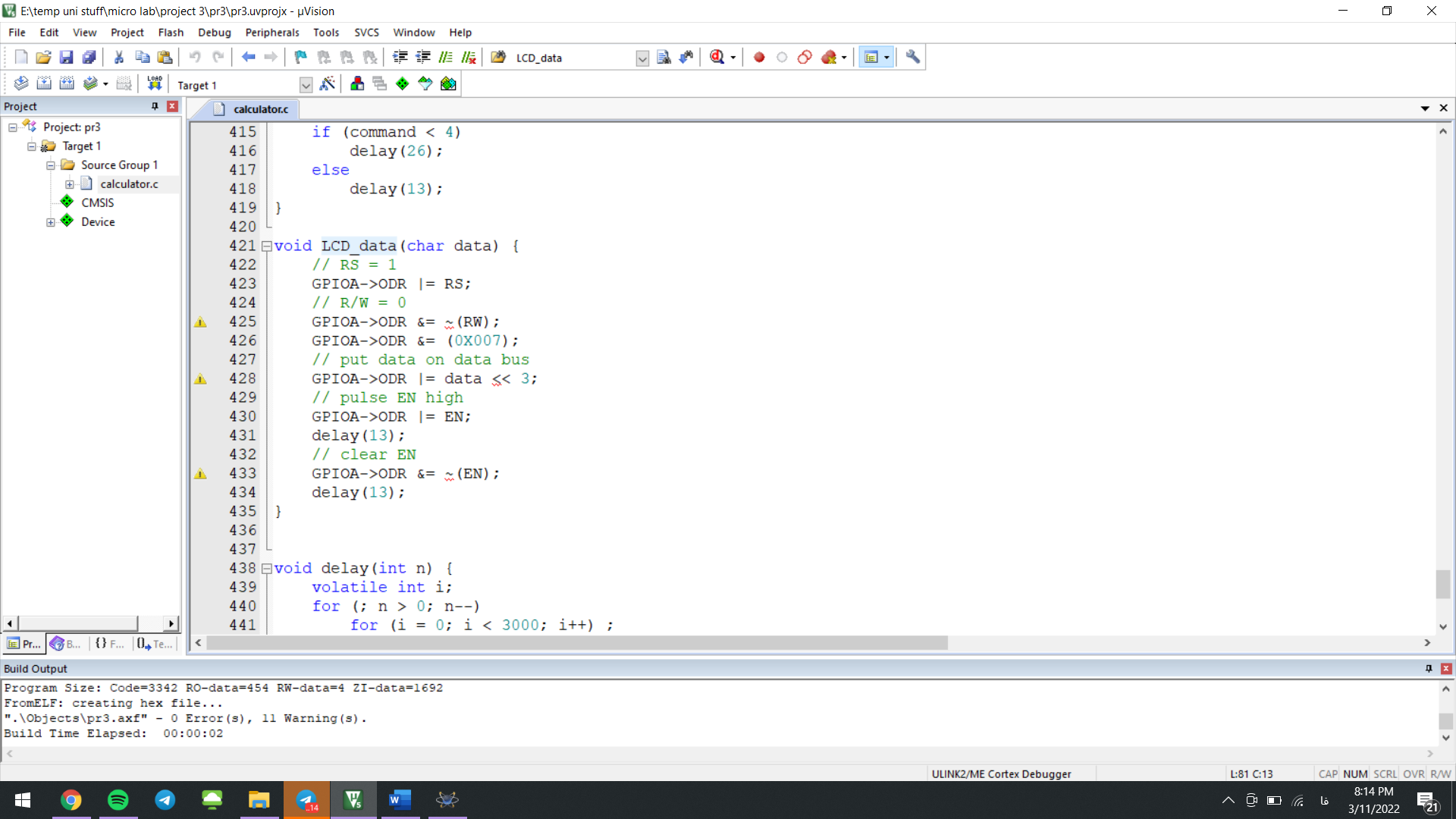
با توجه به نوع عملیات، آن را آنجام داده و در result میریزیم و سپس با تابع lcd\_display آن را نمایش میدهیم.

برای چاپ کردن دو روتین داریم یک روتین هرچه ورودی بگیرد را چاپ می‌کند (lcd\_data) و دیگری برای نمایش اعداد است که در ادامه دومی آمده و سپس اولی.



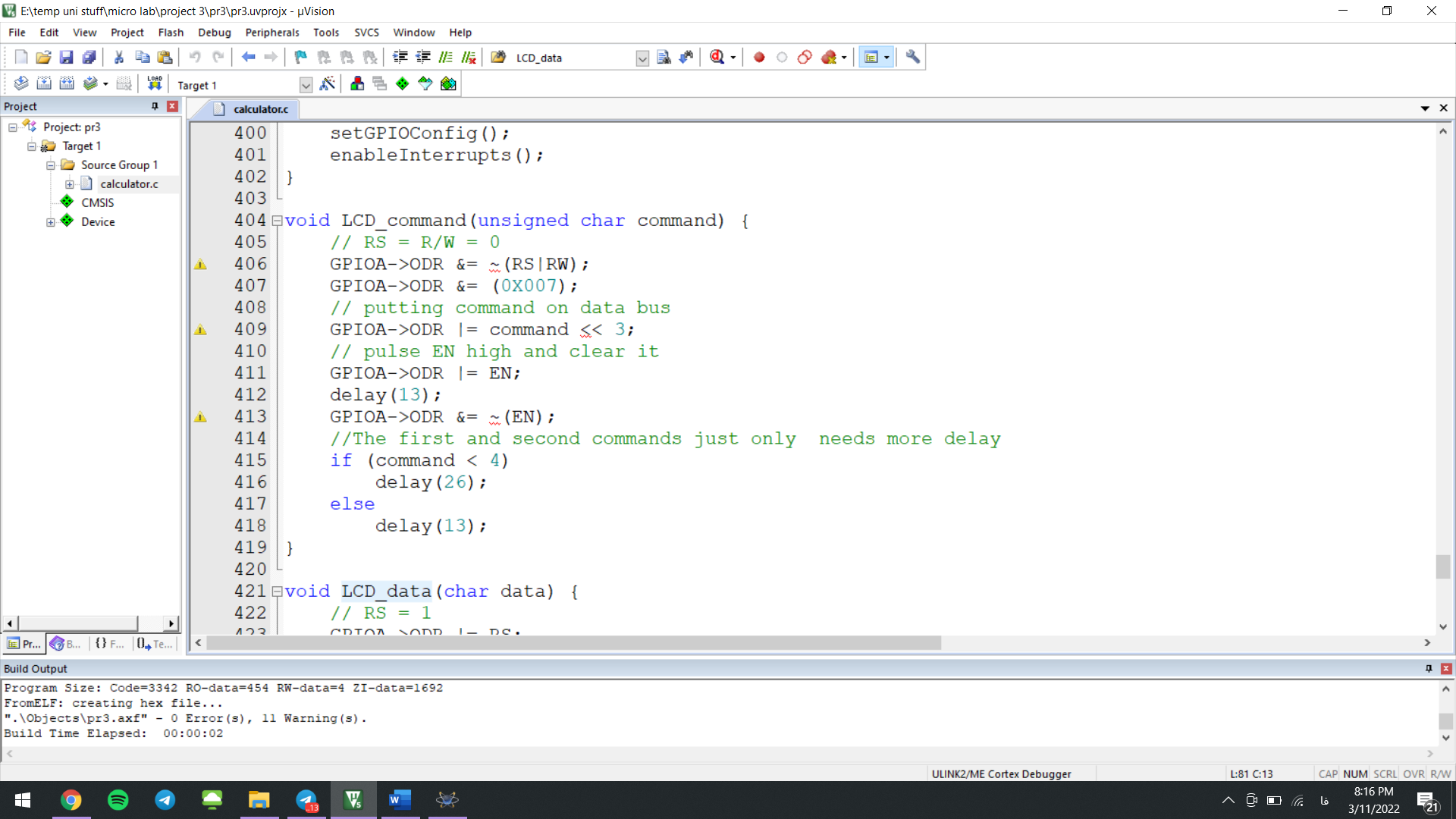
در روتین display\_LCD به کمک جدول زیر ابتدا کامند پاکسازی صفحه نمایشگر صدا زده می شود و سپس عدد ورودی بعنوان پارامتر را نمایش می دهیم

. روتین اول هم data\_LCD است که هرچه ورودی بگیرد چاپ می کند و ما از آن برای نمایش حروف و کاراکتر ها بهره جسته ایم.

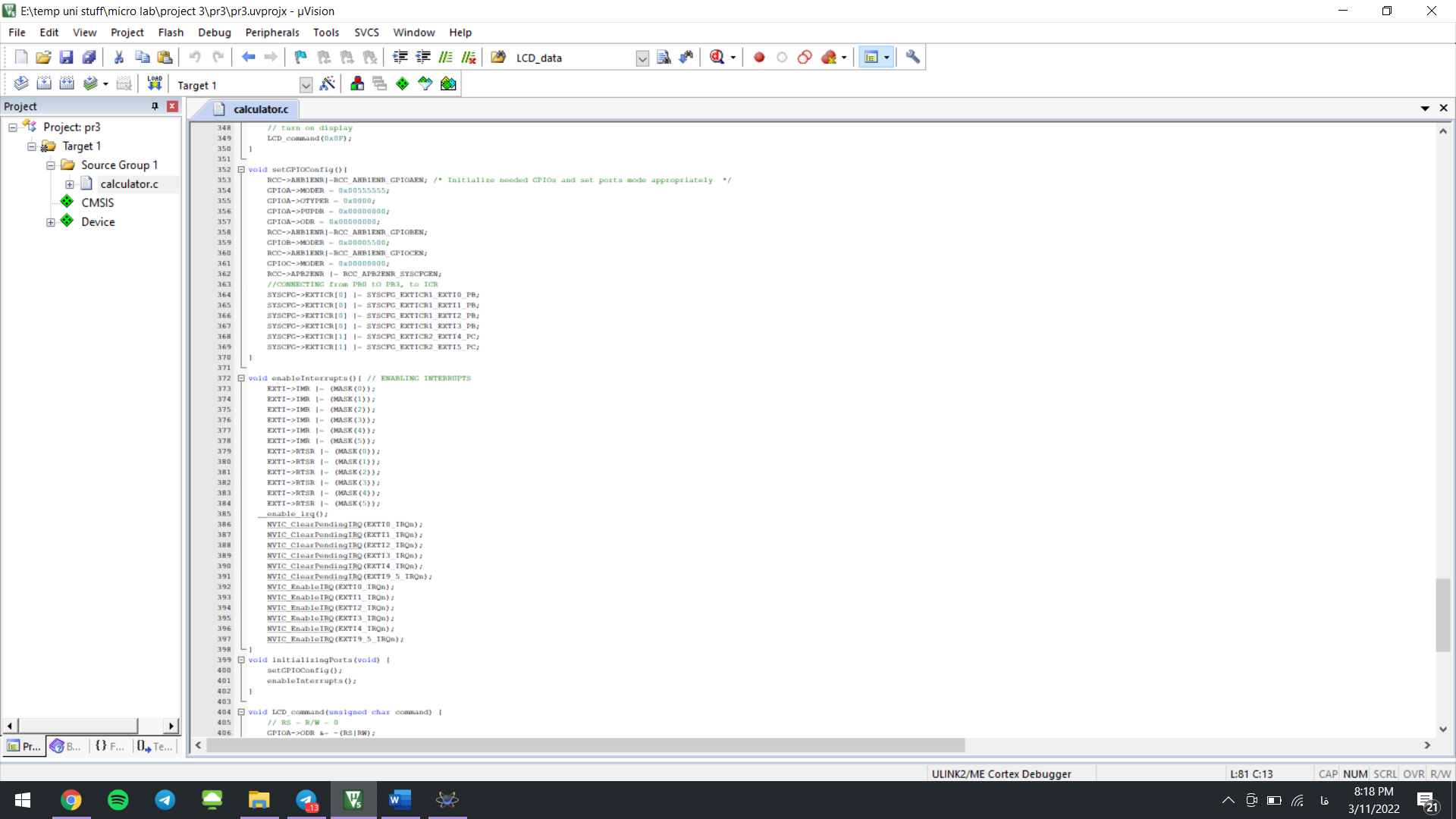


روتین دیگری داریم برای ارسال کامند های طبق جدول کامند های زیر

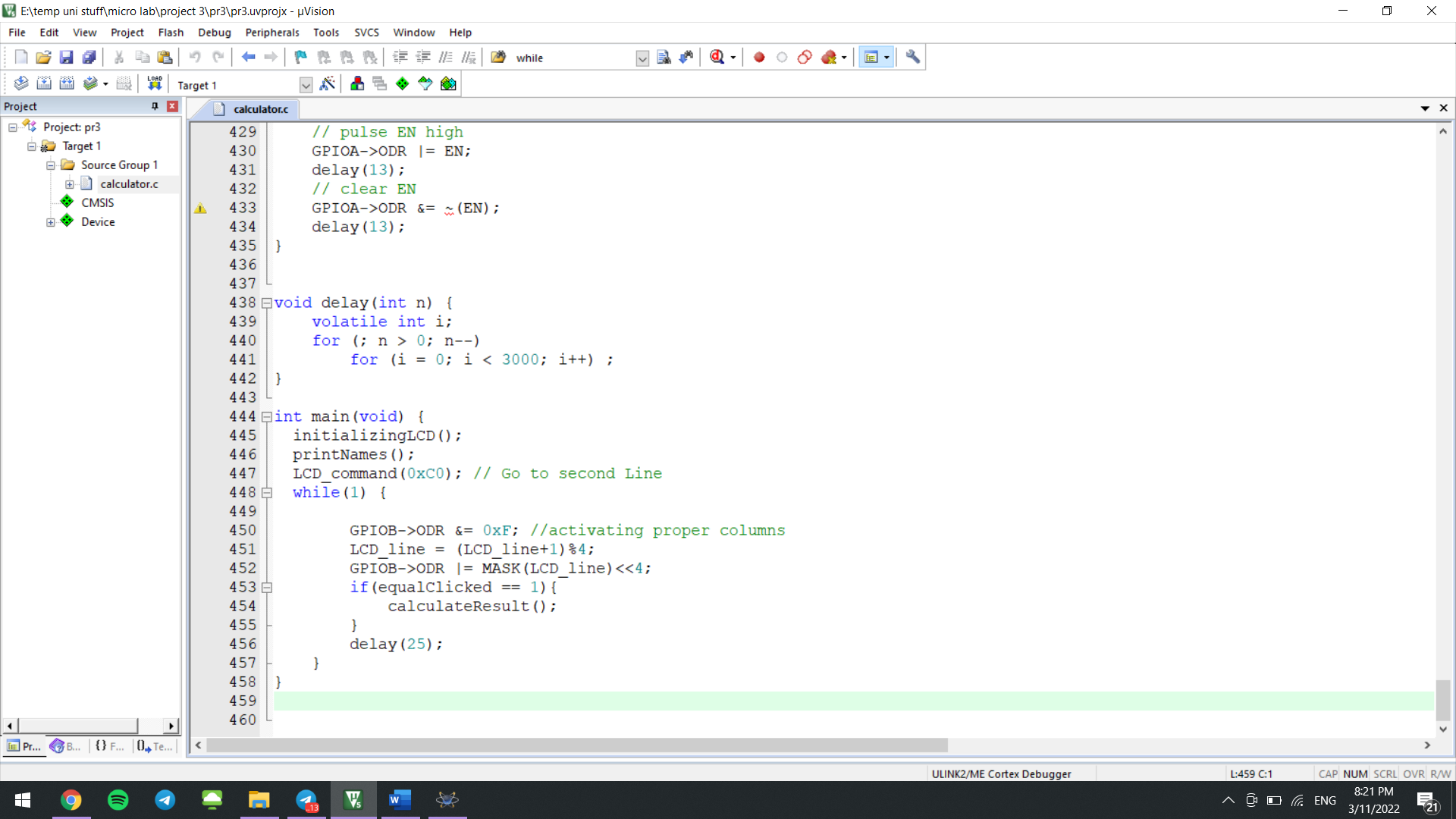




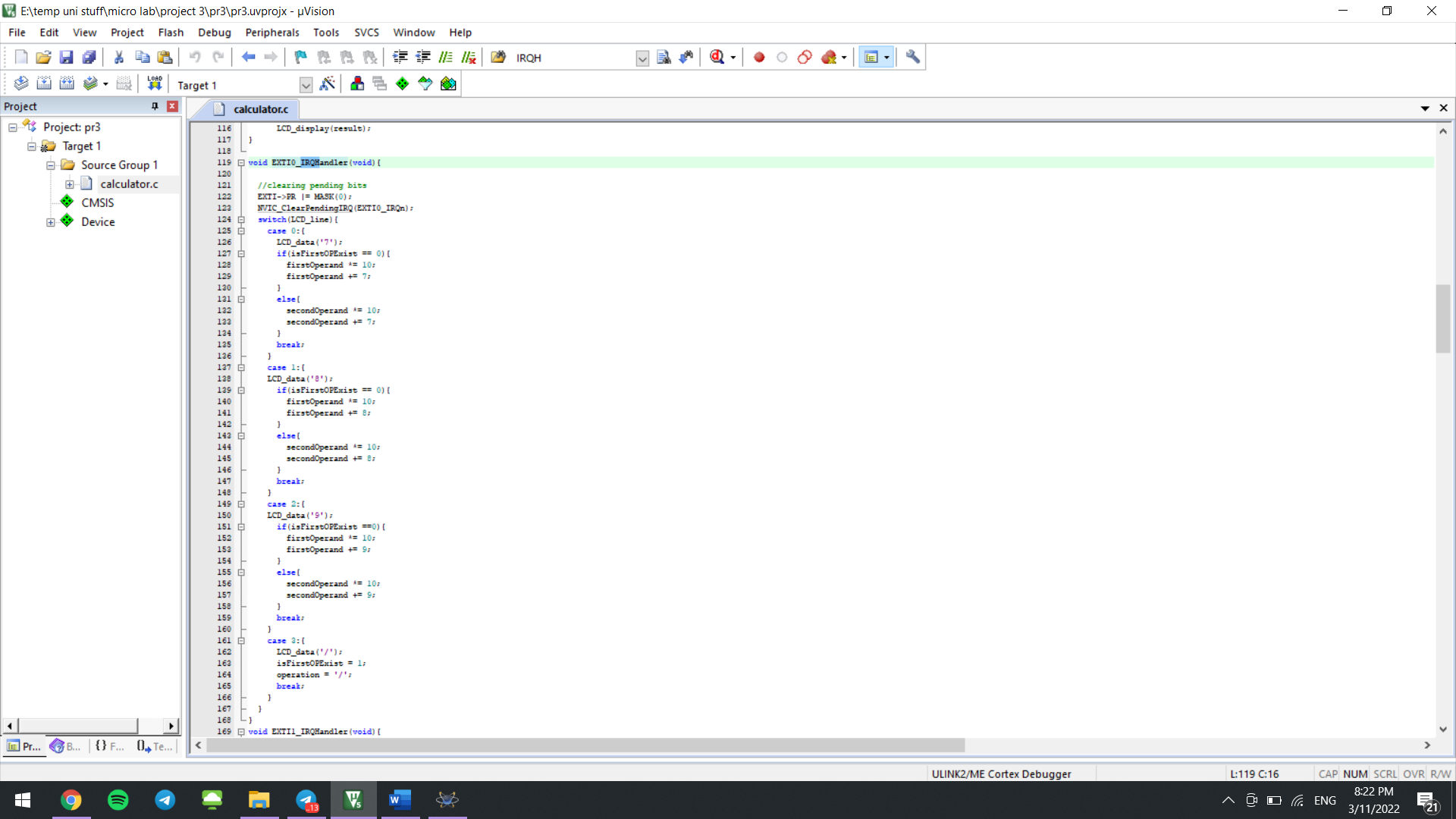
مقداردهی اولیه به پورت ها:



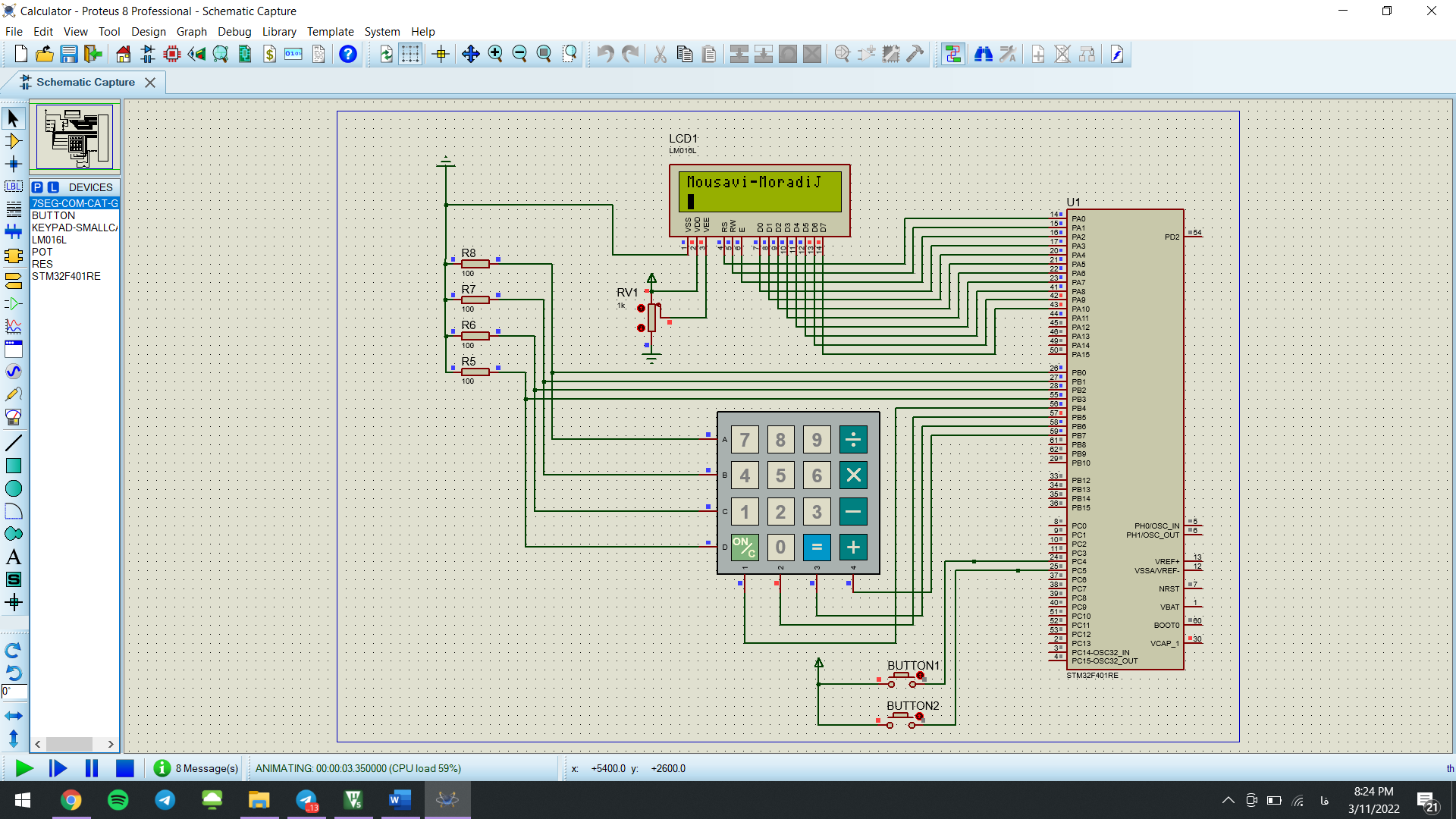
هر سری با کد زیر یکی از خطوط عمودی صفحه کلید فعال می شوند )طوری تنظیم شده سرعت فعال/غیرفعالسازی که با یکبار فشرده شدن اینتراپت ایجاد شود(

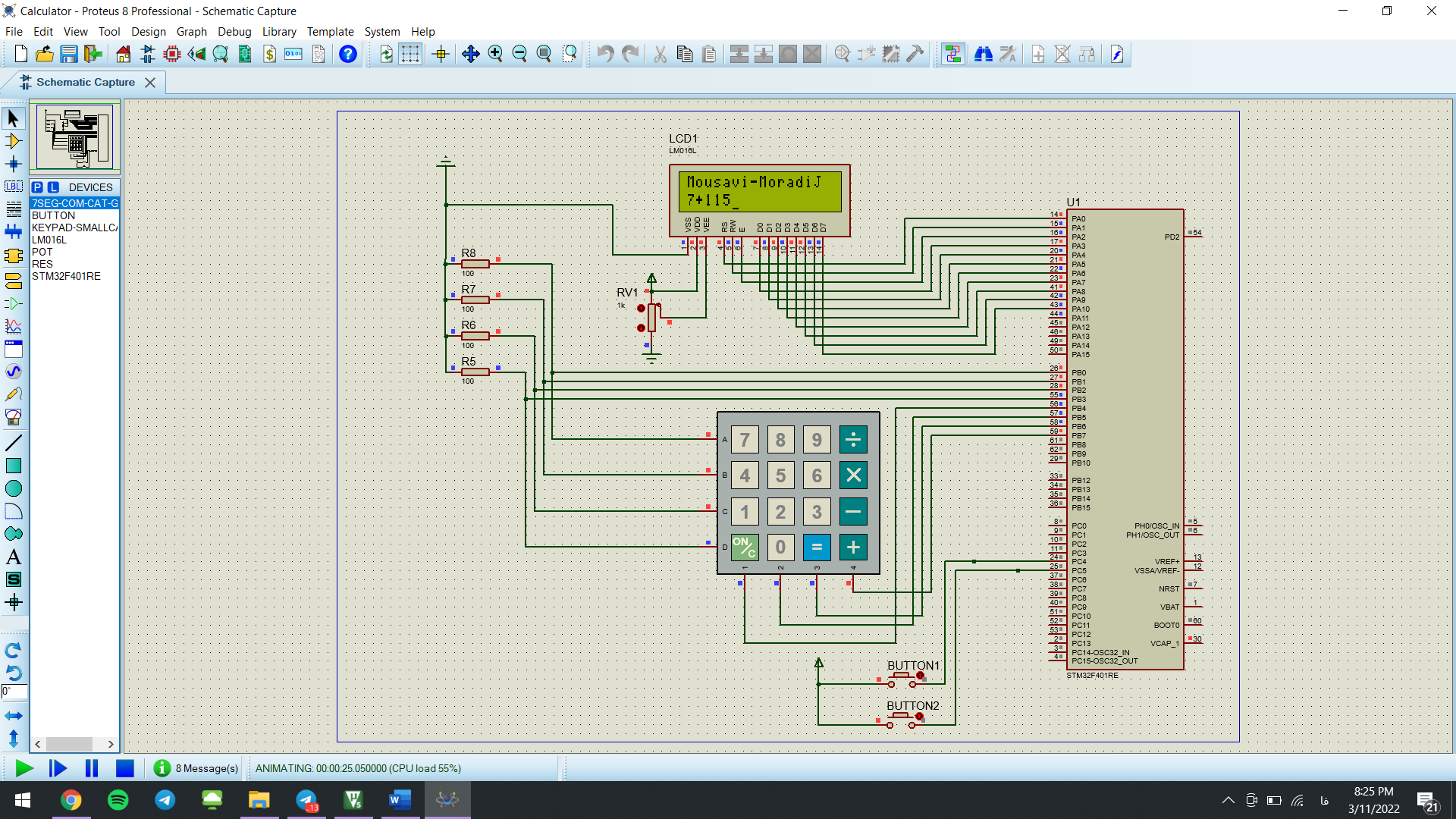


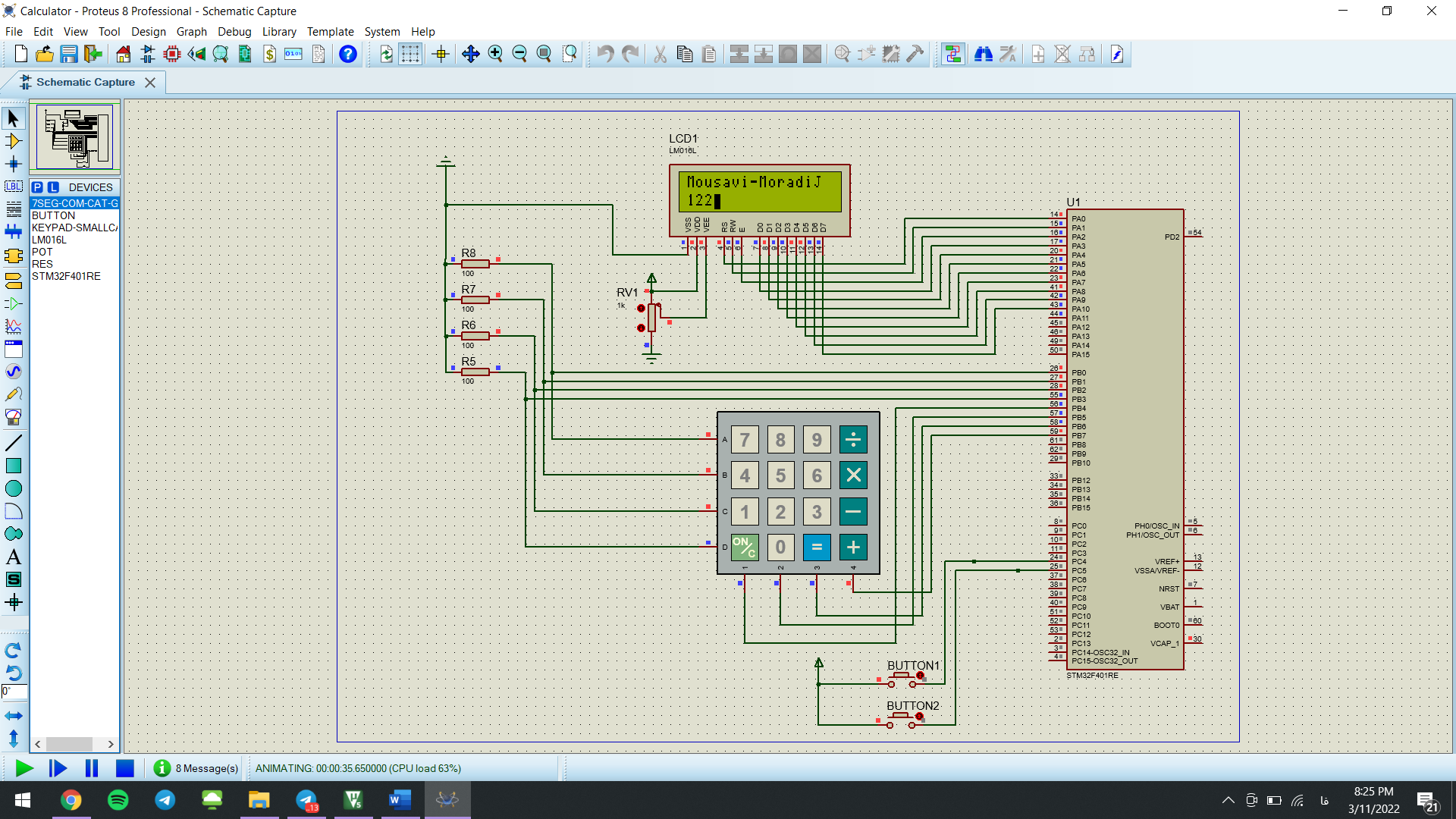
و بسته به اینکه کدام سطر )یکی از کلید های این سطر( فشرده شده به روتین interrupt مختص ان می رویم و در آنجا می فهمیم که کدام کلید فشرده شده به طور مثال اگر یکی از کلید های سطر اول را بفشاریم وارد روتین زیر می شویم.



در ادامه نتیجه کار را در پروتئوس مشاهده می‌کنیم:









**رفرنس دستور کار:**

کلاس درس و اسلاید های درسی

دیتا شیت و رفرنس منوآل