به نام خدا



درس سیستم‌های نهفته

پروژه  
  
شایان صالحی  
ایمان محمدی  
آرمین ثقفیان

عنوان پروژه، پریز هوشمند (پروژه 20)

مقدمه

در عصر حاضر، جایی که تکنولوژی و اینترنت اشیاء (IoT) نقش بسزایی در بهبود کیفیت زندگی ما دارند، هوشمندسازی منزل و محیط کار به یک نیاز اساسی تبدیل شده است. پروژه‌ی «پریز هوشمند» با هدف ارتقاء امکانات خانه هوشمند و افزایش کارایی و انعطاف‌پذیری در کنترل دستگاه‌های الکتریکی طراحی و پیاده‌سازی شده است. این پروژه با استفاده از ماژول کوییکتل و یک رله، امکان کنترل از راه دور دستگاه‌های الکتریکی را از طریق وب یا اپلیکیشن موبایل فراهم می‌کند.

در این پروژه، ما به ساخت یک پریز هوشمند پرداخته‌ایم که قادر است به واسطه‌ی فناوری GSM و اتصال به اینترنت، دستورات کاربر را از هر جای دنیا دریافت کرده و عملیاتی نظیر روشن یا خاموش کردن و یا برنامه‌ریزی زمانی برای فعالیت دستگاه‌های متصل شده را اجرا کند. این امر با استفاده از قابلیت Open CPU موجود در ماژول کوییکتل محقق شده که اجازه می‌دهد بدون نیاز به میکروکنترلر یا پردازنده کمکی، برنامه‌های کاربردی مختلفی را بر روی خود ماژول اجرا کنیم.

قابلیت Open CPU این امکان را به ما می‌دهد که با نوشتن کد مستقیماً بر روی ماژول، ویژگی‌های پیچیده‌تری نظیر برقراری ارتباط با سرورهای MQTT برای دریافت و ارسال پیام، ذخیره‌سازی داده‌ها در حافظه خود قطعه برای ذخیره کاربران، و کنترل دقیق‌تر رله‌ها و سایر قطعات الکترونیکی را بدون واسطه و با دقت بالا انجام دهیم.

هدف

هدف از این پروژه، ایجاد یک پریز هوشمند قابل اعتماد و کاربردی است که کاربران بتوانند به صورت از راه دور و به آسانی دستگاه‌های برقی خود را مدیریت کنند. این شامل برنامه‌ریزی زمان‌های روشن و خاموش شدن پریز از طریق وب یا اپلیکیشن است.

قطعات استفاده شده

* *ماژول M65 نصب شده بر روی بورد*
* *مبدل USB به TTL*
* *رله و سایر قطعات الکتریکی برای مکانیزم سوئیچینگ*

توضیح فنی

این پروژه با استفاده از قابلیت GSM ماژول و ویژگی open CPU آن، برنامه‌ریزی شده است تا دستورات دریافتی از طریق پروتکل MQTT را پردازش کند و عملیاتی نظیر روشن یا خاموش کردن پریز را بر اساس برنامه‌های تعریف شده توسط کاربر اجرا کند..

پیاده‌سازی

در ابتدا برای پیاده‌سازی هرکد مورد نیاز روی این بخش، نیاز داریم که کدهای مورد نیاز را کامپایل کنیم. این کار توسط MS-DOS تعبیه شده در SDK مربوطه انجام می‌شود که مسیر target آن کامپایلر مربوط به آن است. پس از اجرای برنامه مدنظر ما در main.c در پوشه custom، یک فایل با فرمت

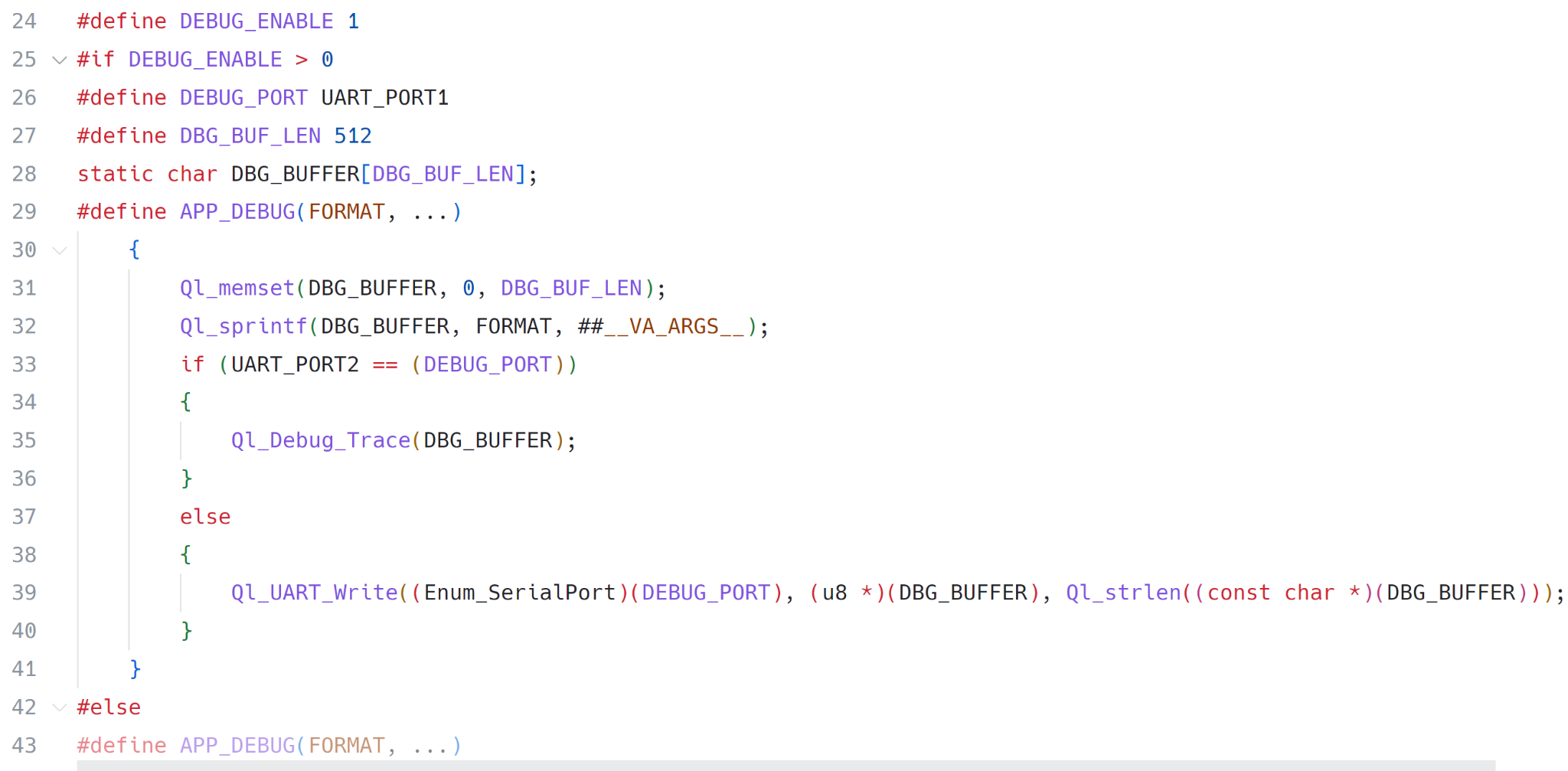
APPGS5MDM32A01.lod ساخته شده که با لود کردن آن در نرم‌افزار QFlash با استفاده از پورت‌های دیباگ، قطعه را پروگرام می‌کنیم.

برای دریافت log های مربوط به قطعه از نرم‌افزار QNavigator بهره گرفتیم که در برنامه اصلی با دستور APP\_DEBUG می‌توان پیام‌های مدنظر را از طریق UART و از طریق پورت‌های RXD و LXD انتقال دهیم.

بخش اول: تعریف و مقدمات

تعریف ماکروها و متغیرها

کد با تعریف ماکروها و متغیرها شروع می‌شود که شامل پورت‌های UART برای ارتباط و دیباگ، طول بافر برای ذخیره داده‌های دریافتی و ارسالی، و پین‌های مربوط به کنترل رله‌ها می‌شود. همچنین، ساختار داده‌ای برای نگهداری وضعیت خروجی‌ها (رله‌ها) تعریف می‌شود.



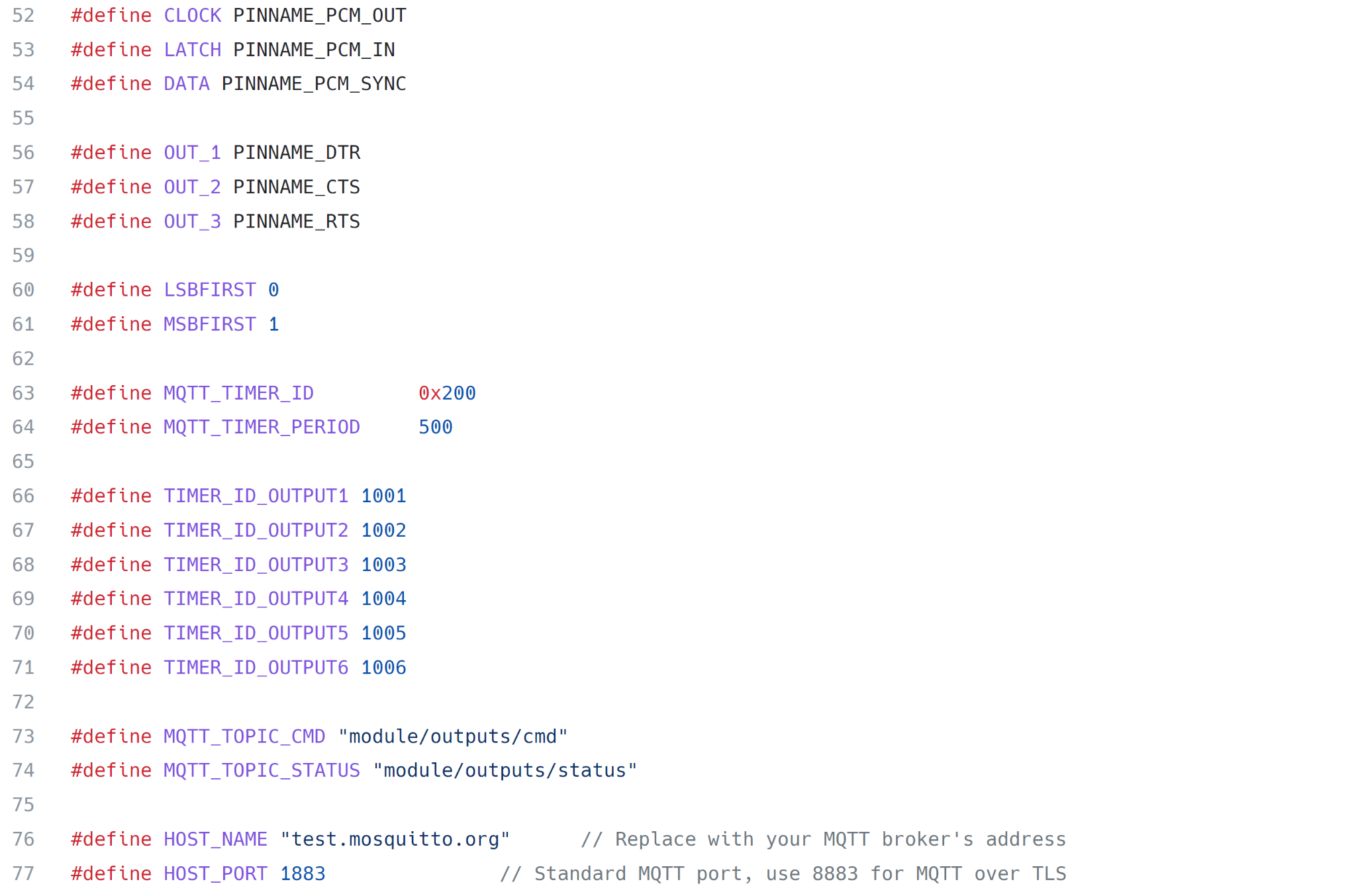
در این قسمت مانند تمامی example ها متغییرهای مربوط به لاگ انداختن، UART و دستور APP\_DEBUG تعریف شده است که در طول پروگرم‌ما نقش اساسی برای نشان دادن استیت‌های مختلف قطعه دارد و همچنین این امکان را فراهم می‌کند تا پیام‌های دیباگ و داده‌های دریافتی از MQTT یا SMS را بخوانیم.

بخش دوم: تعریف متغیرهای اساسی

در این بخش تمامی متغیرهای اساسی همچون چراغ‌های قطعه، تایمرهای مورد نیاز برای پروتکل MQTT و چک کردن استیت آنها و یک متغیر TIMER\_ID منحصر به فرد برای تمامی خروجی‌های که بتواند به صورت موازی و مستقل مکانیزم زمان‌دهی آن را هندل کند.

همچنین درنهایت در اینجا برای ایجاد پروتکل MQTT یک HOST NAME و یک پورت در نظر گرفته شده است که ما آنها را دامین test.mosquitto.org و پورت 1883 استفاده کرده‌ایم.

تصویر مربوط به این قسمت را در صفحه بعد می‌توانید مشاهده کنید.



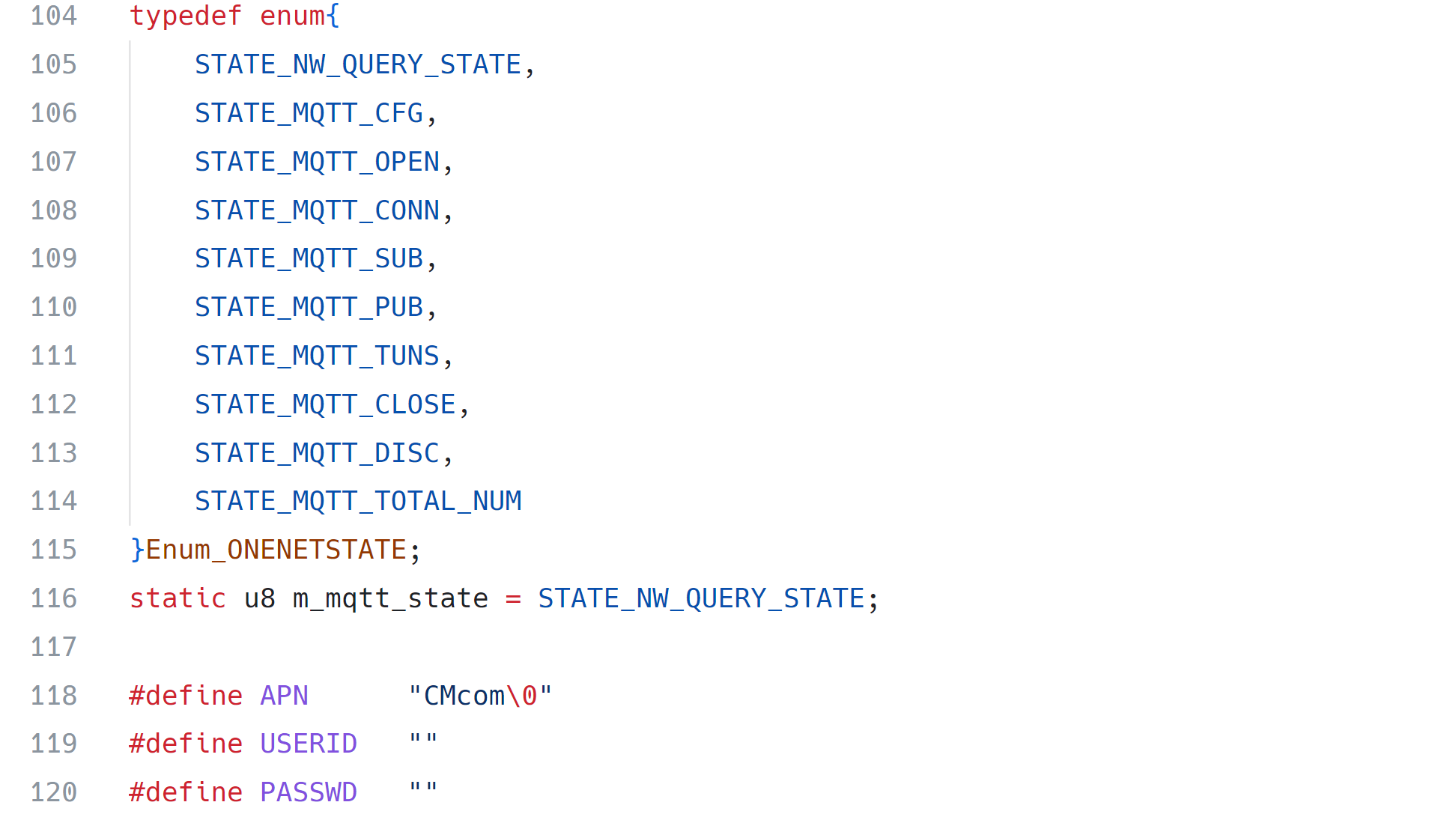
بخش سوم: کار به صورت استیت برای پروتکل MQTT

برای پیاده‌سازی پروتکل MQTT یک تابع اصلی تحت عنوان Callback\_Timer داریم که تمامی استیت های پروتکل MQTT را مدیریت می‌کند.

در ابتدا در استیت دریافت کوئری هستیم و پس از آن می‌بایست با توجه به تنظیمات و پارامترهای داده شده کانفیگ صورت گیرد. پس از آن یک MQTT با هاست و پورت داده شده باز شده و پس از آن متصل می‌شود.

بعد از انجام‌ این‌کارها قطعه‌ما در تایپک از قبل تعیین شده "MIc Outputs" سابسکرایب کرده و پیام‌های رد و بدل شده تحت این موضوع را به عنوان دستور می‌پذیرد. پس از این استیت وارد استیت STATE\_MQTT\_PUB شده و یک پیام از قبل تعیین شده در این فضا publish می‌کند. در نهایت وارد استیت دریافت پیام و خواندن کامند از آن می‌شویم که با توجه به تایمر تعیین شده این تابع صدا زده شده که بررسی کند که آیا پیامی دریافت شده است یا نه.

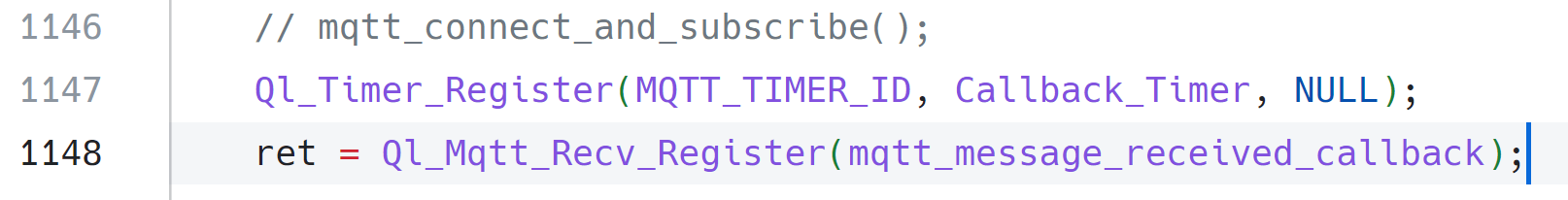
این قسمت تعریف تمامی استیت‌ها بوده:



و این قسمت ساختار کلی تابع طراحی شده را نشان می‌دهد:



که در اینجا شاهد این هستیم که خواندن این تابع به عنوان یک تایمر ست شده است:



بخش چهارم: تابع اصلی

در تابع اصلی که تحت عنوان proc\_main\_task می‌شناسیم در ابتدا سریال پورت‌ها initialize شده سپس تمام قسمت‌های اصلی دیگر همانند وضعیت سیم‌کارت، تایمرها، استک تایمر، چک‌کردن فایل برای بررسی کاربرهای دارای دسترسی و غیره انجام می‌شود.

پس از آن یک حلقه اصلی داریم که همانند زیر همواره پیام‌های دریافتی را بررسی می‌کند:



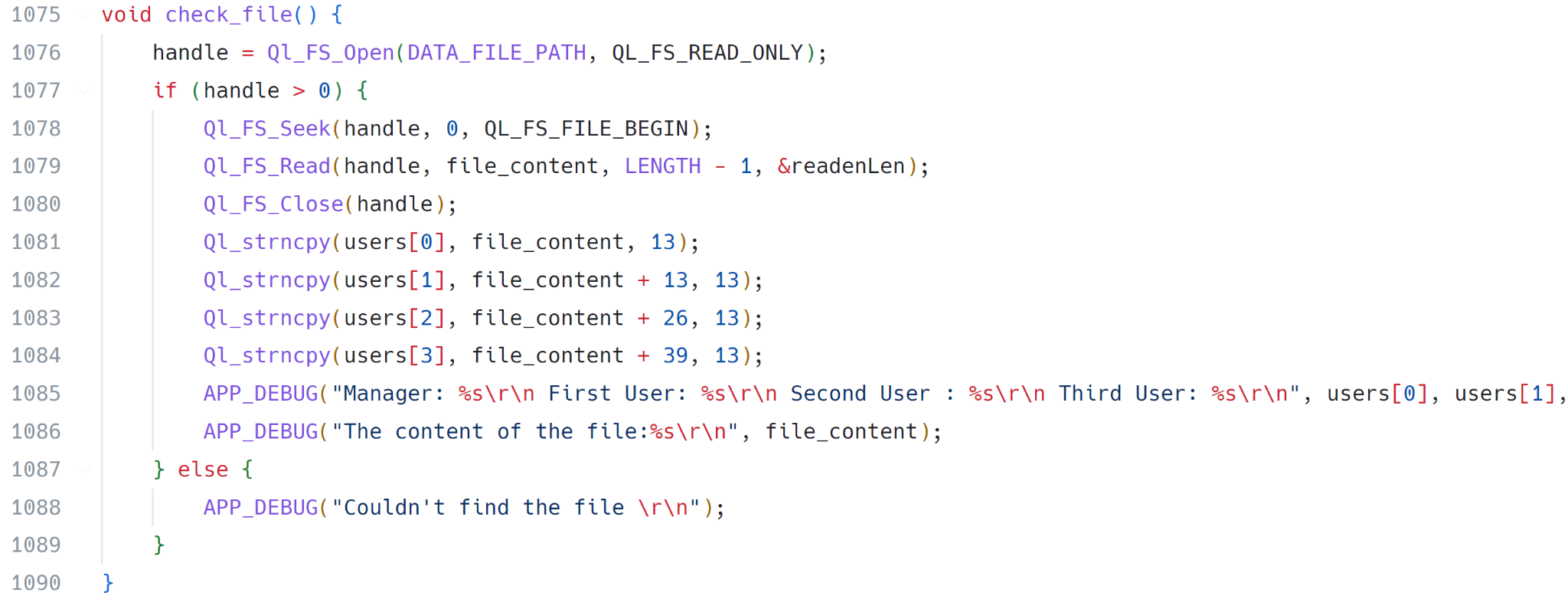
در اینجا در ابتدا تمامی استیت‌های مختلف چک می‌شود که پیام دریافتی از طرف os قطعه چه حالتی دارد. در صورتی که مربوط به پیامک بوده وارد آن استیت‌های شده و در غیر این صورت قسمت‌های MQTT را مدیریت می‌کند. این حلقه به ما اجازه می‌دهد که همزمان پیام‌های SMS و MQTT را دریافت و براساس دستورهای داده شده، action های مورد نظر را انجام دهد.

در صورتی که در حالت URC\_NEW\_SMS\_IND باشیم دستورهای جدید از طریق پیامک دریافت شده و برای دریافت پیام‌های MQTT از تابع زیر استفاده کرده‌ایم:



بخش پنجم: خواندن فایل از روی حافظه تراشه

در اینجا فایل ذخیره شده برای شماره تمامی کاربرها خوانده شده و تمامی user های که برنامه حق دارد از آنها دستور بگیر را می‌توانیم در اینجا تعیین کنیم.



همچنین با استفاده از دستور ADD +98[Phone Number]0 می‌توان کاربر جدید به این مجموعه اضافه نمود.

بخش ششم: تابع پارسر کامند‌ها و انجام دستورات

در اینجا تابعی برای پارس کردن تمامی دستورات و انجام‌ها طراحی شده است که دستورها شامل این بخش‌ها می‌شود:

* *ALL ON -> Turn all on*
* *ALL OFF -> Turn all off*
* *ON OUT [number] -> On out [number] 1 < [number] < 6*
* *OFF OUT [number] -> OF out [number]*
* *Manager -> Make that phone number manager*
* *Get ALL Users -> All the users that have permission to command*
* *Add [Phone number] -> Add +989138094457*
* *Get Outputs -> Get the state of all outputs*
* *Current time -> Get the time of M65*
* *UNTIL [time] ON OUT [number] -> Until given time set output [number] state to on*
* *FOR [seconds] ON OUT [number] -> For amount of [seconds] keeps out [number] on*

که ۶ خروجی درنظر گرفته شده که ۳ تای از آنها مربوط به چراغ‌های LED خود قطعه و سه‌تای دیگر مربوط به خروجی‌های IO بوده است.

در اینجا می‌توانید نحوه بررسی تمامی کامندها را مشاهده کنید:

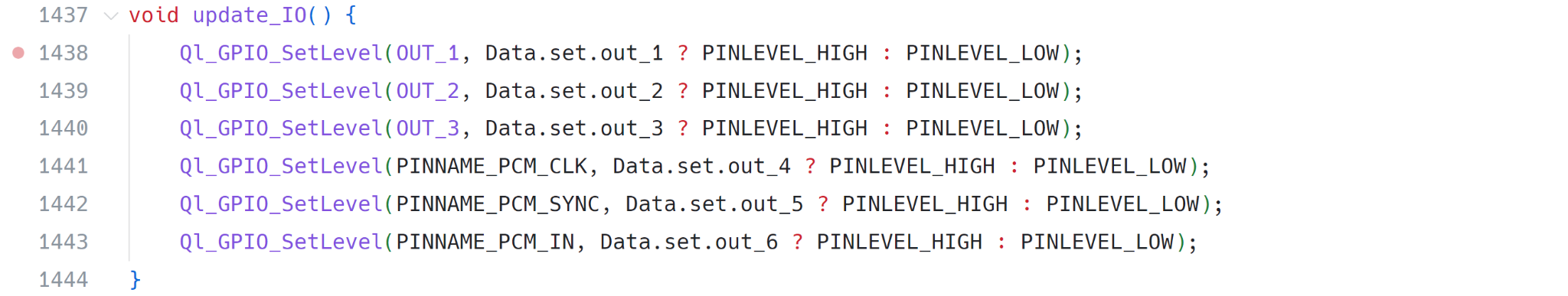


نکته حائز اهمیت در این قسمت این است که برای پابلیش کردن یک پیام در MQTT یا ارسال یک پیامک به همان شماره فرستنده از دو تابع زیر بهره می‌بریم:





و تابع زیر را پس از بررسی تمامی دستورها صدا می‌زنیم که استیت تمامی خروجی‌های را آپدیت کند:

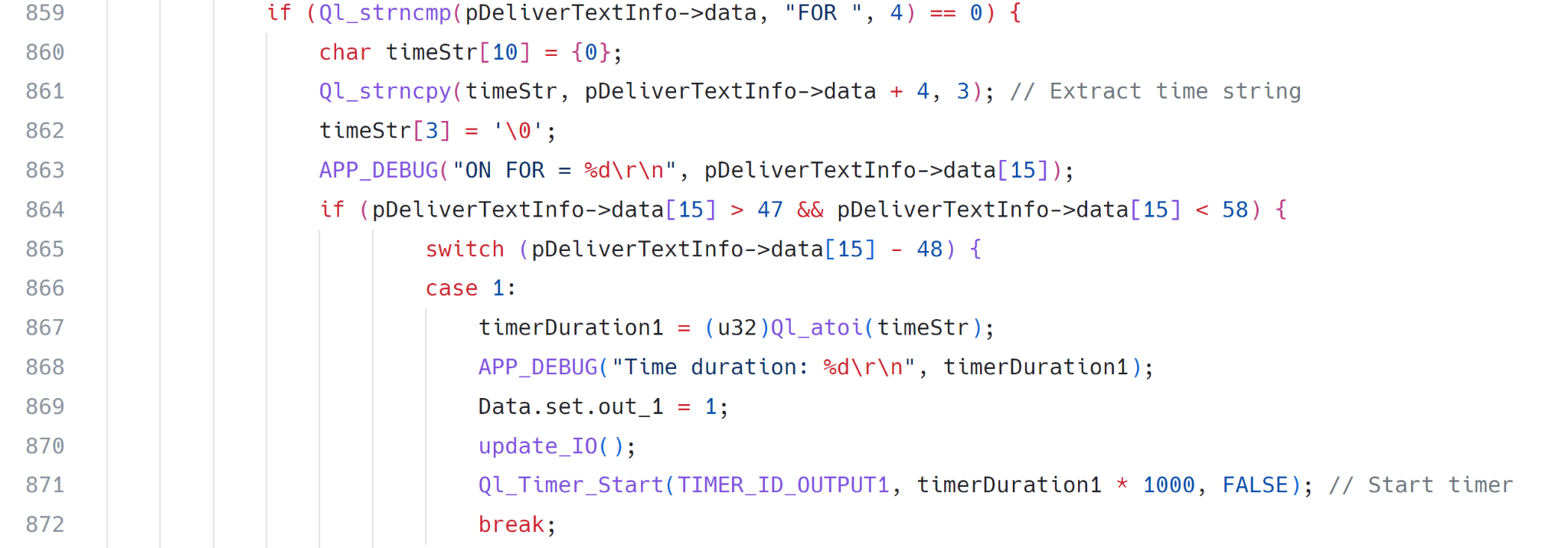


بخش هفتم: بخش تایمر

در اینجا دو مکانیزم تایمر یکی تحت عنوان duration و دیگری تحت عنوان stack پیاده‌سازی شده است. برای روش زمانی از کامند FOR [second]s طراحی شده و برای حالت استک از دستور

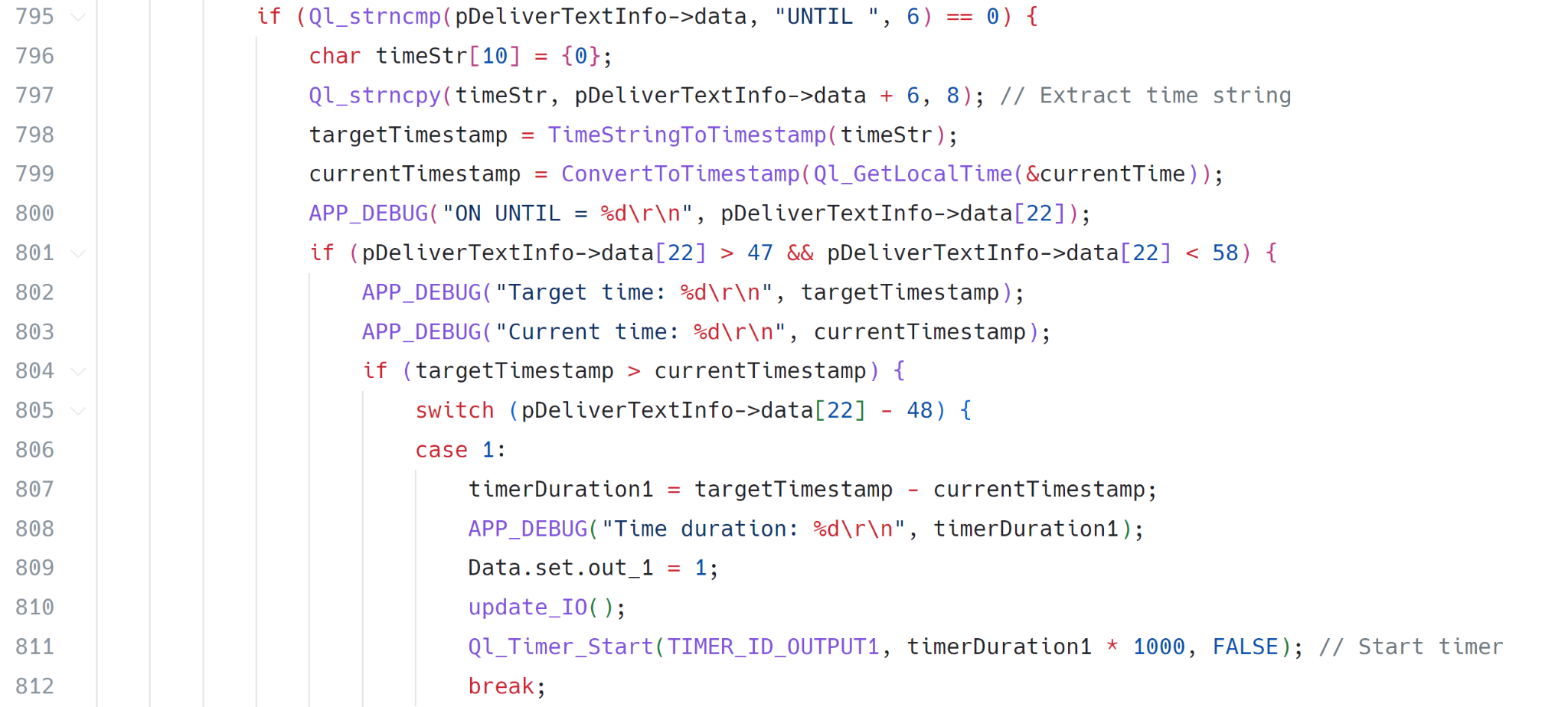
UNTIL [HH:mm:ss] ON استفاده کرده‌ایم.

بخش زیر برای دستور FOR طراحی شده است:

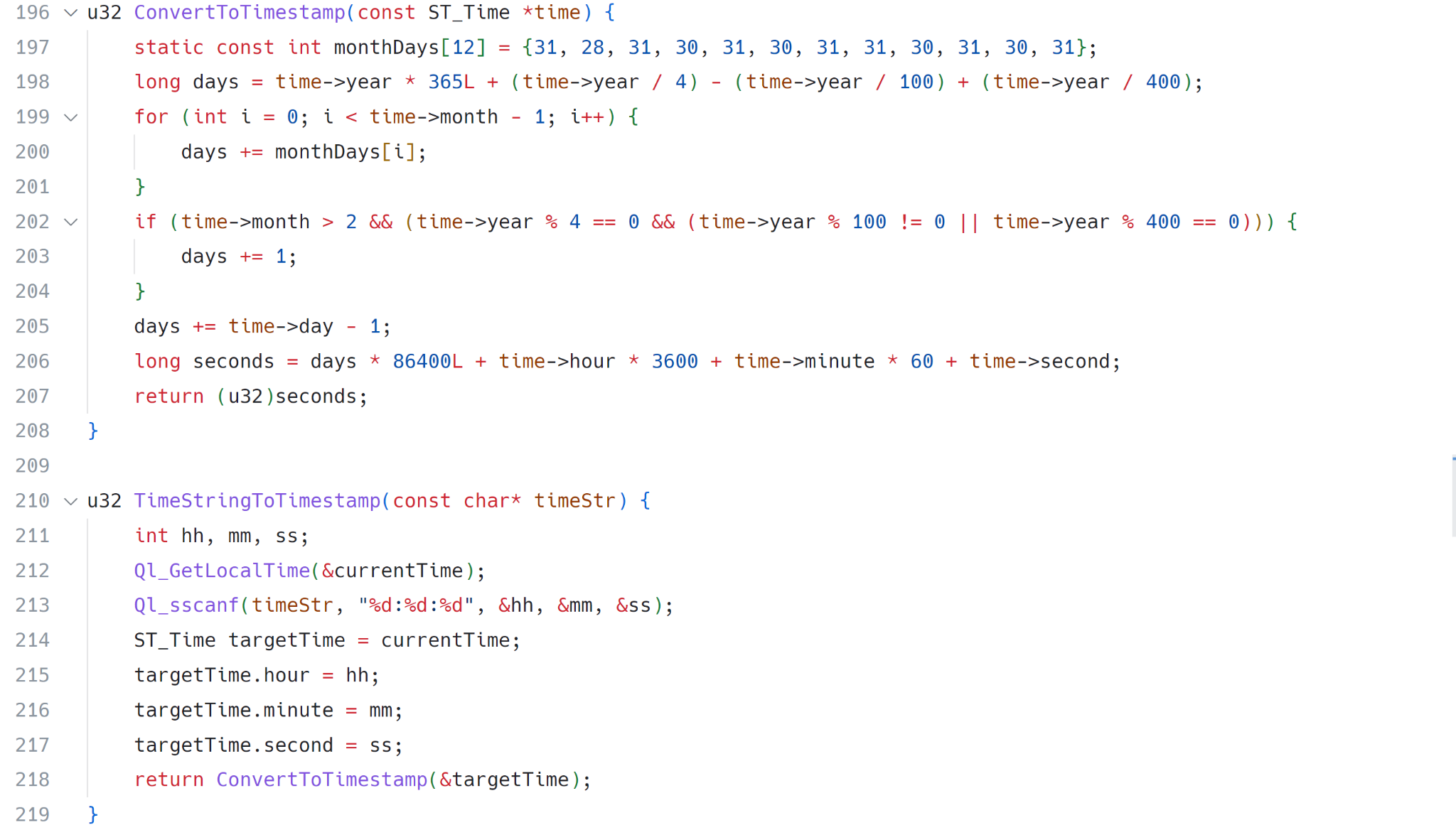


که در آن زمان مدنظر (به صورت ۳ رقم) از کامند برداشته شده و یک تایمر بر اساس آن تنظیم می‌شود.

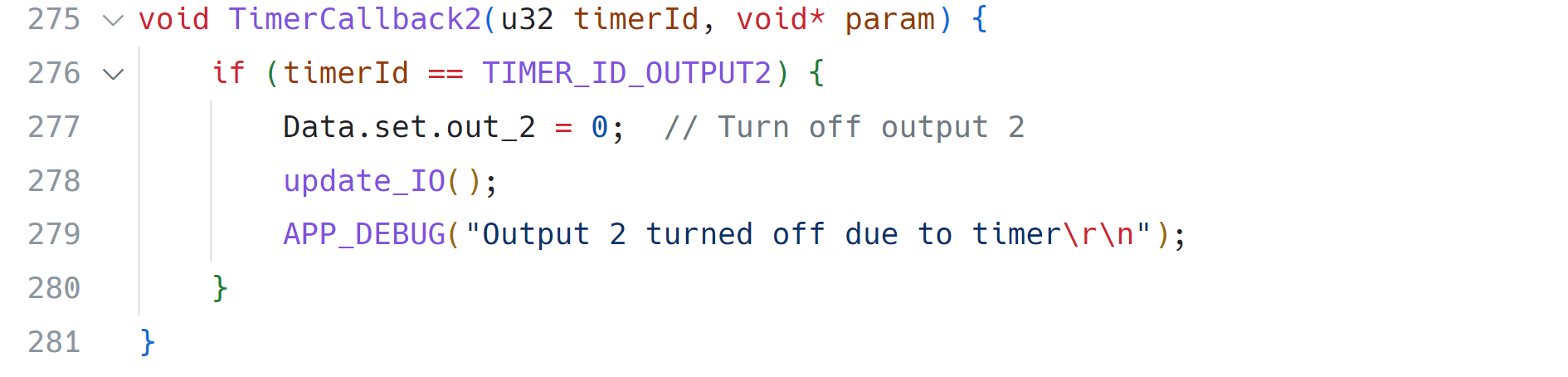
همچنین برای بخش UNTIL به این صورت داریم:



که در آن زمان داده شده در دستور براساس فرمت HH:mm:ss دریافت کرده و براساس تابعی که در ادامه می‌توان دید به فرمتی درمی‌آید که به ثانیه بوده و با کم کردن از زمان فعلی قطعه می‌توان تعیین کنیم که یک خروجی تا چه زمانی روشن بماند.



همچنین تابع زیر زمانی صدا زده می‌شود که تایمر به انتها می‌رسد:

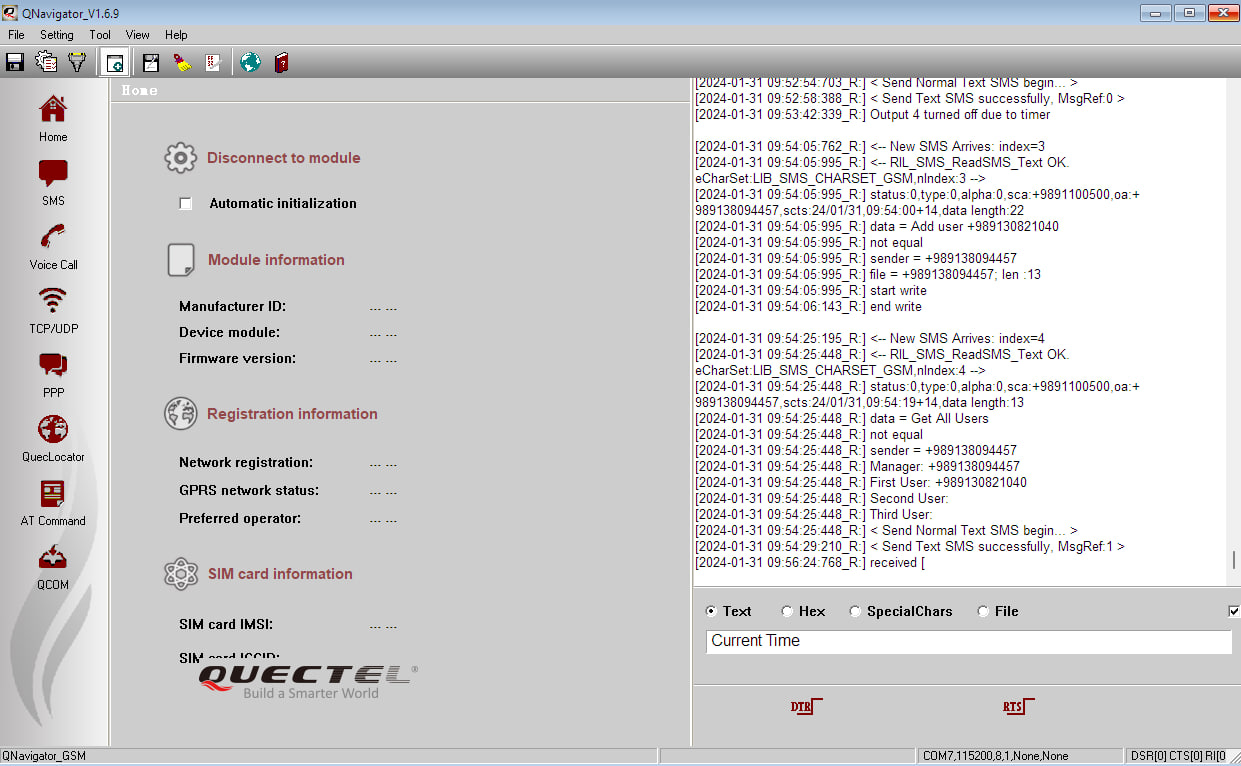


که به ازای هر خروجی یک تابع منحصر به فرد تعریف شده است و پس از بررسی شرط اصلی آن خروجی را دوباره به حالت خاموش می‌برد.

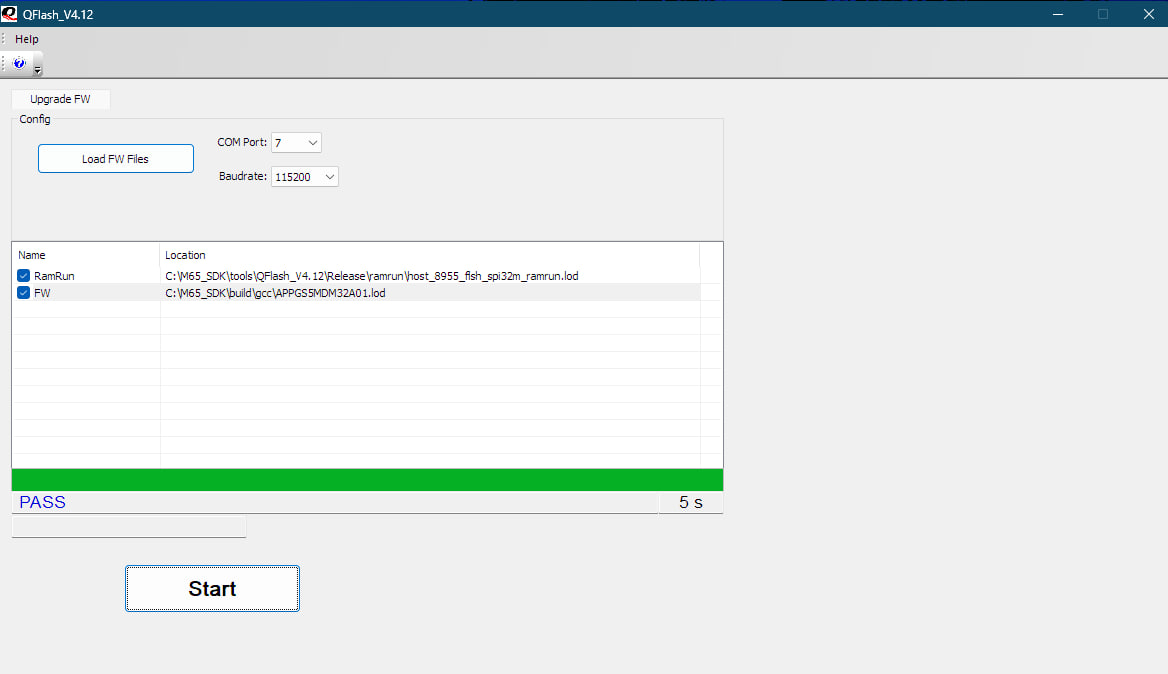
این دو دستور به ما این امکان را می‌دهد که به صورت موازی و مستقل مدت زمان روشن شدن تمامی خروجی‌ها را کنترل کنیم.

بخش هشتم: نرم‌افزار‌های استفاده شده در طول انجام پروژه

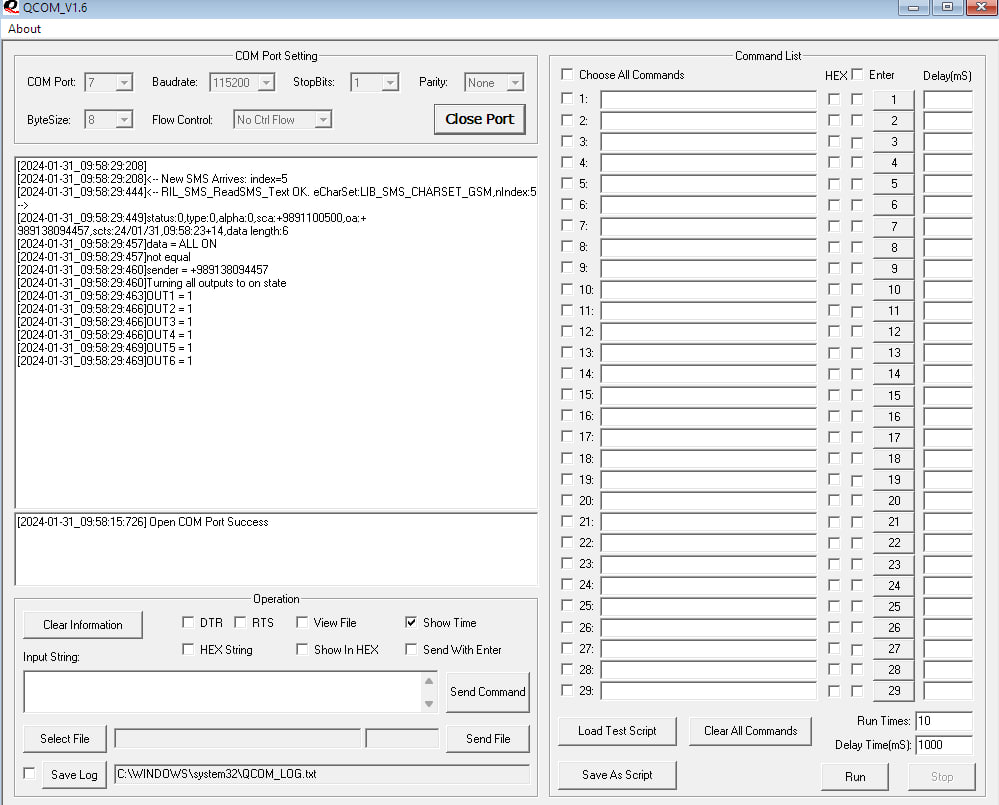
نرم‌افزار QNavigator:



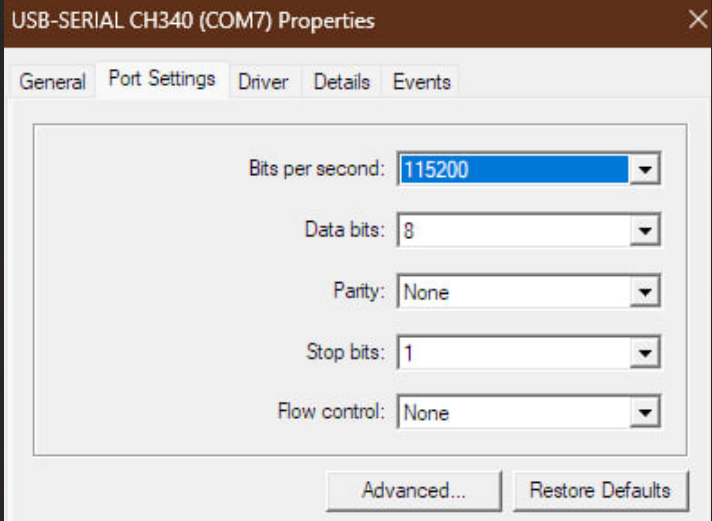
نرم‌افزار QFlash:



نرم‌افزار QCOM:



تنظیمات سریال مربوط به قسمت UART:



خلاصه:

در این پروژه که هدف اصلی آن کار با ماژول M65 شرکت کوییکتل بود که توانستیم با استفاده از SMS و پروتکل MQTT دستورها را به ماژول داده و از راه دور بتوانیم یک خروجی یک پریز را به همراه زمان‌بندی آن کنترل کنیم. چالش‌های اصلی این پروژه آشنایی و یادگیری با نحوه پروگرام کردن ماژول و استفاده از تمامی example های موجود در SDK مربوطه به قطعه، برای طراحی برنامه‌ای که بتواند به صورت همزمان از دستورهای پیامکی و MQTT پشتیبانی کنید و همچنین قابلیت تایمر و استفاده از حافظه خود ماژول را داشته باشد.

همچنین با بکارگیری نرم‌افزارهای همچون QNavigator، QCOM توانستیم تمامی لاگ‌های مربوط به ماژول را بررسی کرده و در طی تکمیل کردن کد پروگرام آن، تمامی حالت‌های دارای اشکال را رفع کرده و برنامه‌ای بی‌نقص برای این ماژول ارائه دهیم.